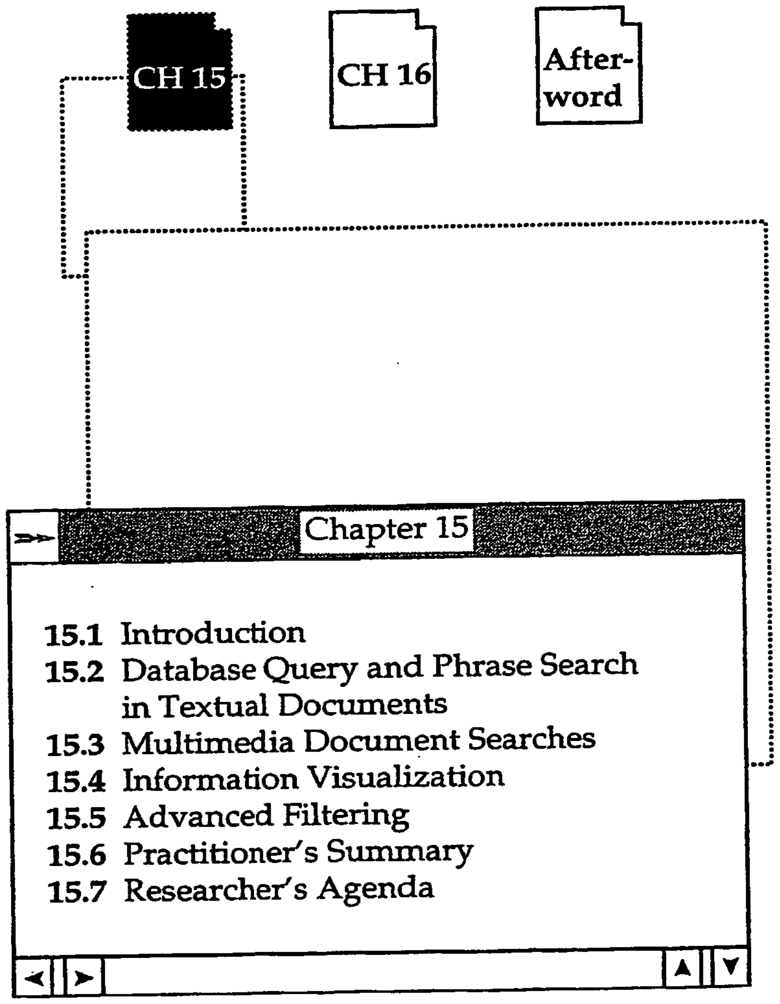
CHAPTER

15

Information Search and Visualization

Everything points to the conclusion that the phrase "the language of art" is more than a loose metaphor, that even to describe the visible world in images we need a developed system of schemata.

E. H. Gombrich, *Art and Illusion*, 1959 (p. 76)



15.1 Introduction

Information exploration should be a joyous experience, but many commentators talk of information overload and anxiety (Wurman, 1989). However, there is promising evidence that the next generation of digital libraries will enable convenient exploration of growing information spaces by a wider range of users. User-interface designers are inventing more powerful search and visualization methods, while offering smoother integration of technology with task.

The terminology swirl in this domain is especially colorful. The older terms of *information retrieval* (often applied to bibliographic and textual document systems) and *database management* (often applied to more structured relational database systems with orderly attributes and sort keys), are being pushed aside by newer notions of *information gathering, seeking, filtering,* or *visualization.* Business-oriented developers focus on the huge volumes of data when they talk of *data mining* and *warehousing,* while expert-system visionaries talk about *knowledge networks.* The distinctions are subtle; the common goals range from finding a narrow set of items in a large collection that satisfy a well-understood information need (known-item search) to browsing to discover unexpected patterns within the collection (Marchionini, 1995).

Exploring information collections becomes increasingly difficult as the volume and diversity grows. A page of information is easy to explore, but when the information representation becomes the size of a book, or library, or even larger, it may be difficult to locate known items or to browse to gain an overview. The strategies to focus and narrow are well understood by librarians and information-search specialists, and now these strategies are being implemented for widespread use. The computer is a powerful tool for searching, but traditional user interfaces have been a hurdle for novice users (complex commands, Boolean operators, unwieldy concepts) and an inadequate tool for experts (difficulty in repeating searches across multiple databases, weak methods for discovering where to narrow broad searches, poor integration with other tools) (Borgman, 1986). This chapter suggests novel possibilities for first-time or intermittent versus frequent computer users, and also for task novices versus experts. Improvements on traditional text and multimedia searching seem possible as a new generation of visualization strategies for query formulation and information presentation emerges.

Designers are just discovering how to use rapid and high-resolution color displays to present large amounts of information in orderly and user-controlled ways. Perceptual psychologists, statisticians, and graphic designers (Bertin, 1983; Cleveland, 1993; Tufte, 1983, 1990) offer valuable guidance about presenting static information, but the opportunity for dynamic displays takes user-interface designers well beyond current wisdom.

The objects–actions interface (OAI) model (see Fig. 2.2) helps by separating task concepts (do you think of your organization as a hierarchy or a matrix?) from interface concepts (is your hierarchy can best represented as an outline, node–link diagram, or a treemap?). The OAI model also separates high-level interface issues (are overview diagrams necessary for navigation?) from low-level interface issues (will color or size coding be used to represent salary levels?).

First-time users of an information-exploration system (whether they have little or much task knowledge) are struggling to understand what they see on the display while keeping in mind their information needs. They would be distracted if they had to learn complex query languages or elaborate shape-coding rules. They need the low cognitive burdens of menu and direct-manipulation designs and simple visual-coding rules. As users gain experience with the interface, they can request additional features by adjusting control panels. Knowledgeable and frequent users want a wide range of search tools with many options that allow them to compose, save, replay, and revise increasingly elaborate query plans.

To facilitate discussion, we need to define a few terms. *Task objects*, such as Leonardo's notebooks or sports-video segments from the Olympics, are represented by *interface objects* in structured relational databases, textual document libraries, or multimedia document libraries. A *structured relational database* consists of *relations* and a *schema* to describe the relations. Relations have *items* (usually called *tuples* or *records*), and each item has multiple *attributes* (often called *fields*), which each have *attribute values*. In the relational model, items form an unordered set (although one attribute can contain sequencing information or be a unique key to identify or sort the other items) and attributes are *atomic*.

A *textual document library* consists of a set of *collections* (typically up to a few hundred collections per library) plus some *descriptive attributes* about the library (for example, name, location, owner). Each collection has a *name* plus some descriptive attributes about the collection (for example, location, media type, curator, donor, dates, geographic coverage), and a set of items (typically 10 to 100,000 items per collection). Items in a collection may vary greatly, but usually a moderate-sized superset of attributes exists that covers all the items. Attributes may be blank, have single values, have multiple values, or be lengthy texts. A collection is owned by a single library, and an item belongs to a single collection, although exceptions are possible. A *multimedia document library* consists of collections of documents that can contain images, sound, video, animations, and so on.

Task actions such as *fact finding* are decomposed into *browsing* or *searching* and are represented by *interface actions* such as scrolling, zooming, joining, or linking. Users begin by formulating their information needs in the task domain. Tasks can range from specific fact finding, where there is a single readily identifiable outcome, to more extended fact finding, with uncertain but replicable outcomes. Relatively unstructured tasks include open-ended browsing of known collections and exploration of the availability of information on a topic:

Specific fact finding (known-item search)

Find the Library of Congress call number of "Future Shock."
Find the telephone number of Bill Clinton.
Find the highest-resolution LANDSAT image of College Park at noon on Dec. 13, 1997.

Extended fact finding

What other books are by the author of Jurassic Park?
What genres of music is Sony publishing?
Which satellites took images of the Persian Gulf War?

Open-ended browsing

Does the Mathew Brady Civil War photo collection show the role of women in that war?
Is there new work on voice recognition being reported from Japan?
Is there a relationship between carbon-monoxide levels and desertification?

Exploration of availability

What genealogy information is available at the National Archives?
What information is available on the Grateful Dead band members?
Do NASA datasets demonstrate acid-rain damage to soy crops?

Once users have clarified their information needs, the first step in satisfying those needs is to decide where to search (Marchionini, 1995). The conversion of information needs, stated in task-domain terminology, to interface actions is a large cognitive step, but it must be accomplished before expression of these actions in a query language or via a series of mouse selections can begin.

Supplemental *finding aids* can help users to clarify and pursue their information needs. Examples include tables of contents or indexes in books, descriptive introductions, concordances, key-word-in-context (KWIC) lists, and subject classifications. Careful understanding of previous and potential search requests, and of the task analysis, can improve search results by allowing the system to offer hot-topic lists and useful classification schemes. For example, the U.S. Congressional Research Service has a list of approximately 80 hot topics covering current bills before Congress, and has 5000 terms in its Legislative Indexing Vocabulary. The National Library of Medicine maintains the Medical Subject Headings (MeSH), with 14,000 items in a seven-level hierarchy.

This chapter covers database query and textual-document searches briefly, then suggests innovative directions for multimedia-document searches and introduces a four-phase framework. The main contribution is a taxonomy of information-visualization strategies based on data types and user tasks. The final section explores advanced filtering methods.

15.2 Database Query and Phrase Search in Textual Documents

Searching in structured relational database systems is a well-established task for which the SQL language has become a widespread standard (Reisner, 1988). Users write queries that specify matches on attribute values, such as author, date of publication, language, or publisher. Each document has values for the attributes, and database-management methods enable rapid retrieval even with millions of documents. For example, an SQL-like command might be

```
SELECT DOCUMENT#
FROM JOURNAL-DB
WHERE   (DATE >= 1994 AND DATE <= 1997)
   AND  (LANGUAGE = ENGLISH OR FRENCH)
   AND  (PUBLISHER = ASIS OR HFES OR ACM).
```

SQL has powerful features, but using it requires training (2 to 20 hours), and even then users make frequent errors for many classes of queries (Welty, 1985). Alternatives such as *query-by-example* can help users to formulate simpler queries, such as requesting all English-language ACM articles published during or after 1994:

```
JOURNAL | DOCUMENT# | DATE   | AUTHOR | LANGUAGE | PUBLISHER
--------------------------------------------------------------
        | P._X      | >=1994 |        | ENGLISH  | ACM
        |           |        |        |          |
```

The full set of Boolean expressions, however, is difficult to express except inside a special *condition box.*

Form-fillin queries can substantially simplify many queries, and, if the user interface permits, some Boolean combinations (usually a conjunction of disjuncts (ORs) within attributes with ANDs between attributes) can be easy to express:

```
JOURNAL DATABASE
  DOCUMENT#:
       DATE: 1994..1997
     AUTHOR:
   LANGUAGE: ENGLISH, FRENCH
  PUBLISHER: ASIS, HFES, ACM
```

Although SQL is a standard, many form-fillin variants for expressing relational database queries have been proposed to aid novice searchers. The diversity is itself an impediment to easy use, but designers assume that users are willing to invest minutes or hours to learn each interface. This assumption is not valid for walk-up kiosks or for web pages offering textual-document library searches, in which users are often invited to type keywords or *natural-language queries* in a box, and to click on a run button. This presentation is meant to be appealing, but the computer's capacity for responding to the natural-language query is often limited to eliminating frequent terms or commands ("please list the documents that deal with") and searching for remaining words. A ranked list of documents is usually presented, and users must do their best in choosing relevant items from the list.

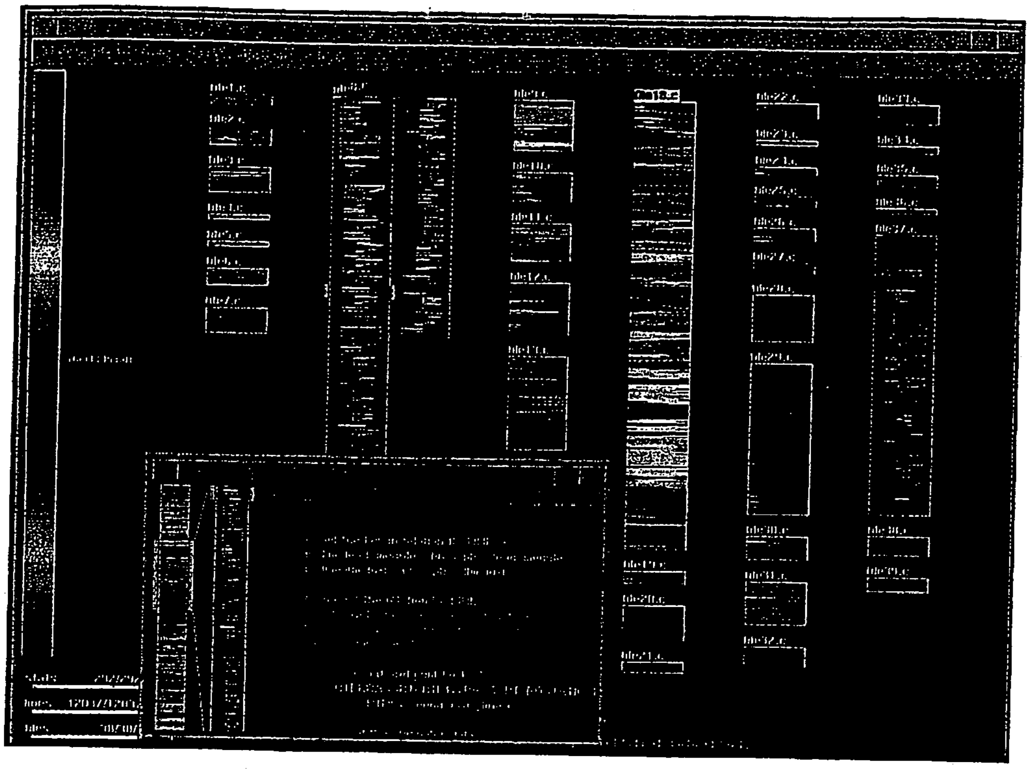


Plate B1: A computer program with 4000 lines of code. The newest lines are in red; the oldest are in blue. The smaller browser window shows a code overview and detail view. (Used with permission of ATT Bell Labs, Naperville, IL.)

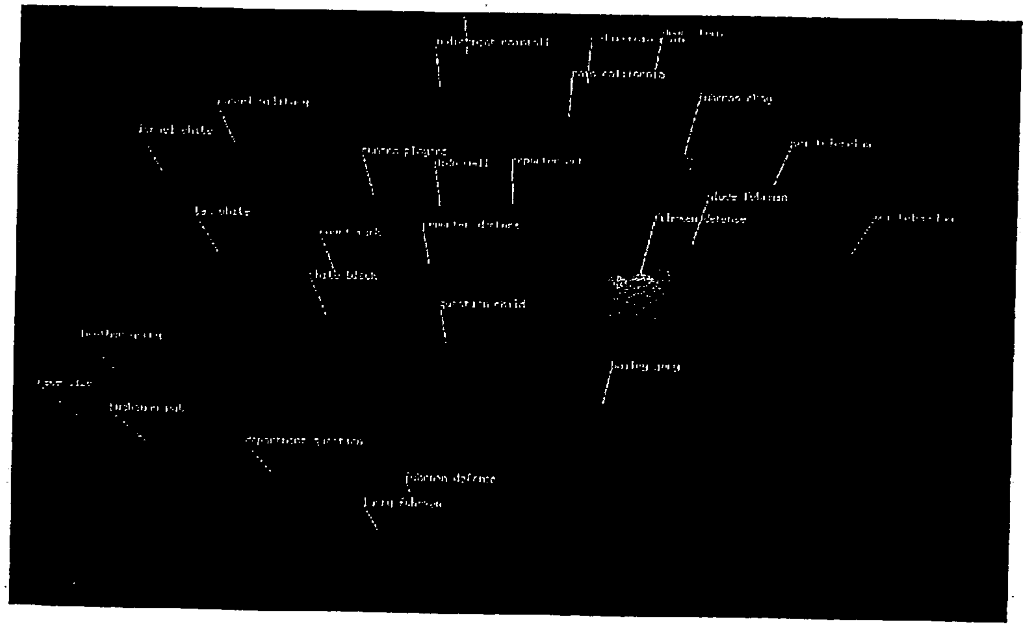
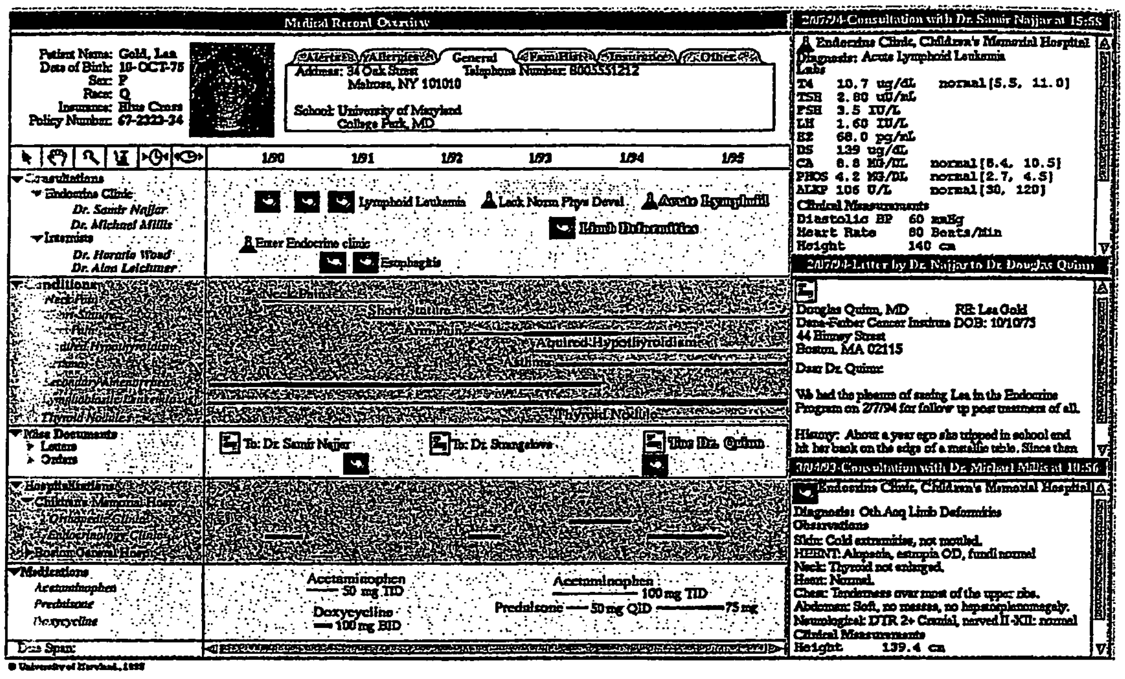


Plate B2: Information-retrieval themescape, showing a multidimensional information space pressed down into a two-dimensional topological map. Some clustering of points can be interesting, but they carry the danger of misinterpretations of the meaning of adjacency. (Wise et al., 1995.) (Used with permission of Battelle Pacific Northwest National Laboratory, Richland, WA.)



2/07/94-Consultation with Dr. Samir Najjar at 15:55

Endocrine Clinic, Children's Memorial Hospital
Diagnosis: Acute Lymphoid Leukemia
Labs

```
T4   10.7 ug/dL    normal[5.5, 11.0]
TSH  2.80 uU/mL
FSH  3.5 IU/L
LH   1.60 IU/L
E2   68.0 pg/mL
DS   139 ug/dL
CA   8.8 MG/DL     normal[8.4, 10.5]
PHOS 4.2 MG/DL     normal[2.7, 4.5]
ALKP 106 U/L       normal[30, 120]
```

Clinical Measurements

```
Diastolic BP   60 mmHg
Heart Rate     80 Beats/Min
Height         140 cm
```

2/07/94-Letter by Dr. Najjar to Dr. Douglas Quinn

Douglas Quinn, MD RE: Lea Gold
Dana-Farber Cancer Institute DOB: 10/10/75
44 Binney Street
Boston, MA 02115

Dear Dr. Quinn:

We had the pleasure of seeing Lea in the Endocrine Program on 2/7/94 for follow up post treatment of all.

History: About a year ago she tripped in school and hit her back on the edge of a metallic table. Since then

3/04/93-Consultation with Dr. Michael Millis at 10:56

Endocrine Clinic, Children's Memorial Hospital
Diagnosis: Oth Acq Limb Deformities
Observations
Skin: Cold extremities, not mottled.
HEENT: Alopecia, esotropia OD, fundi normal
Neck: Thyroid not enlarged.
Heart: Normal.
Chest: Tenderness over most of the upper ribs.
Abdomen: Soft, no masses, no hepatosplenomegaly.
Neurological: DTR 2+ Cranial, nerved II-XII: normal
Clinical Measurements

```
Height       139.4 cm
```

© University of Maryland, 1997

Plate B3: Medical version of LifeLines. Physician visits, conditions, hospitalizations, and medications are shown. Every item in the record is seen as a line or icon in the overview, with color coding by doctor. Line thickness indicates severity and dosage. Windows on the right side show the details. (Plaisant et al., 1997.)

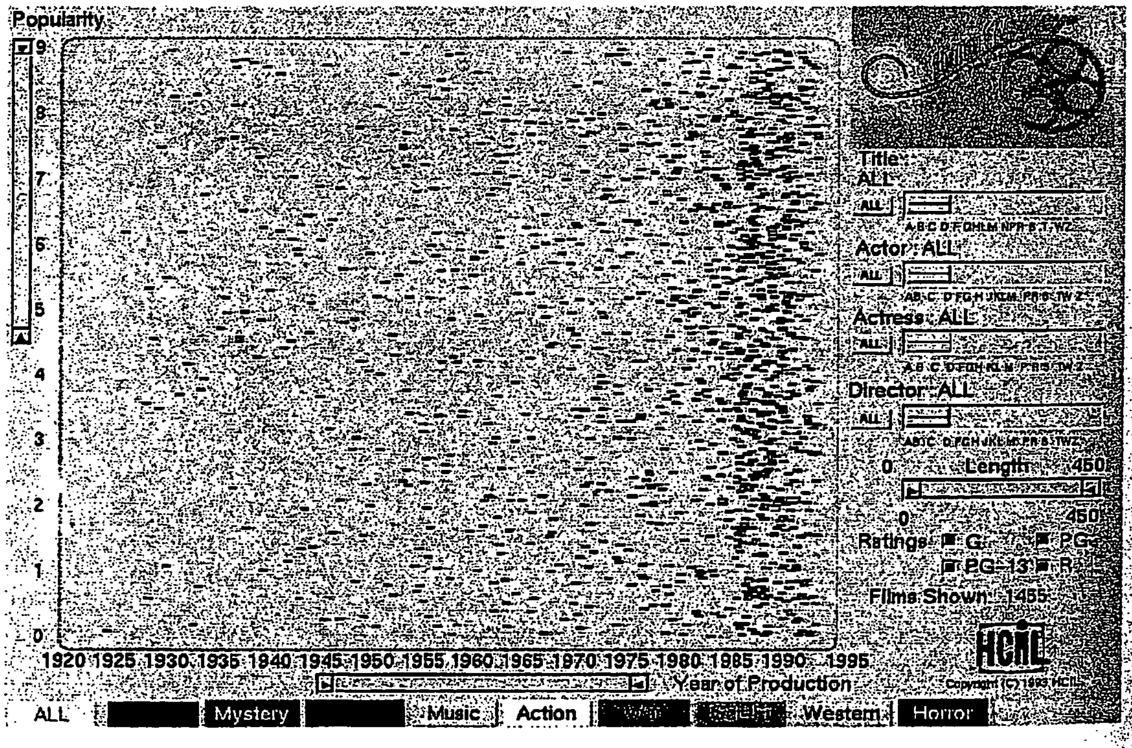


Plate B4 (a): FilmFinder showing 1500 films in a starfield display, where the location of each point is determined by the year of the film (*x* axis) and by the film's popularity in video store rentals (*y* axis). The color encodes the film type.

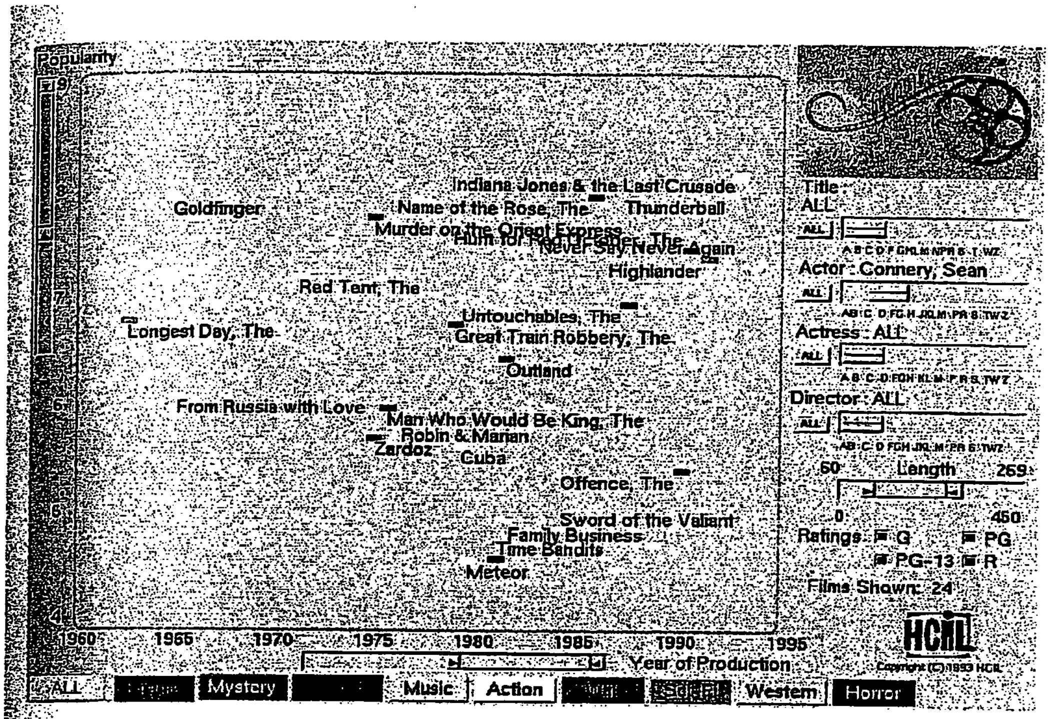
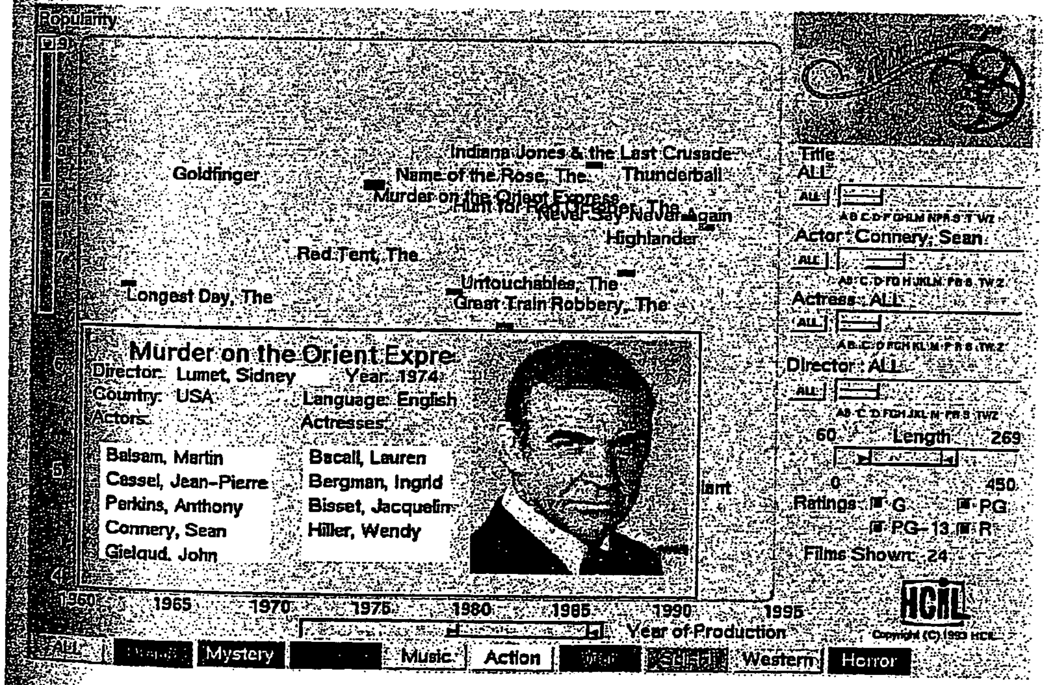


Plate B4 (b): FilmFinder after zooming in on recent popular films. When less than 25 films remain, the titles appear automatically.



Murder on the Orient Expre

Director: Lumet, Sidney Year: 1974

Country: USA Language: English

Actors:	Actresses:
Balsam, Martin	Bacall, Lauren
Cassel, Jean-Pierre	Bergman, Ingrid
Perkins, Anthony	Bisset, Jacquelin
Connery, Sean	Hiller, Wendy
Gielgud, John	

Plate B4 (c): FilmFinder after selection of a single film. The info card pops up with details on demand. (Ahlberg and Shneiderman, 1997.)

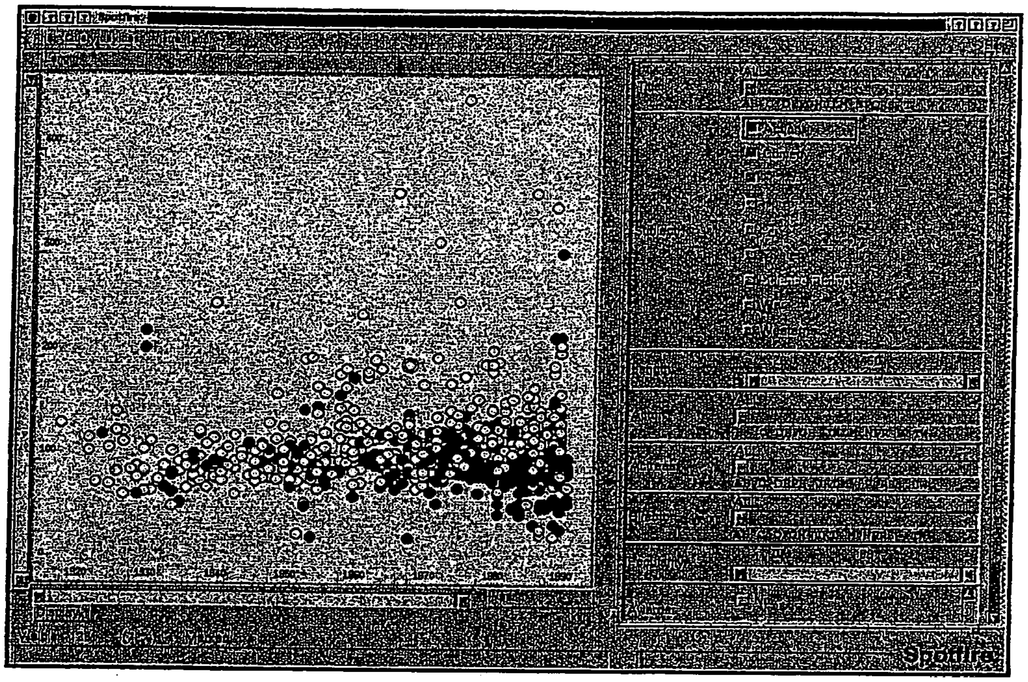


Plate B5: Spotfire version of FilmFinder, which provides increased user controls. Users can set axes (set to length in minutes and year) and glyph attributes (color is set to subject, and larger size indicates award-winning film). (Used with permission of IVEE Development, Goteborg, Sweden.) (http://www.ivee.com)

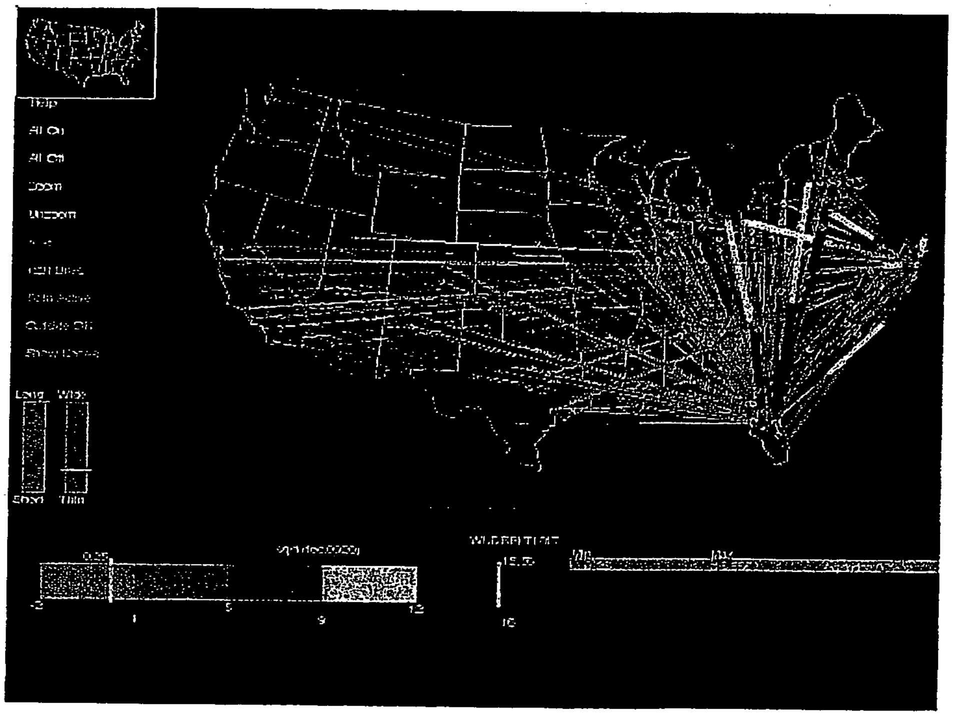


Plate B6: Telephone network traffic represented by thickness and color of the half-line segments between cities. (Used with permission of ATT Bell Labs, Naperville, IL.)

Although an easy-to-use interface is a good idea, if users cannot express their intentions or are uncertain about the meaning of the results, then the interface may need improvement. Finding a way to provide powerful search without overwhelming novice users is a current challenge. Existing interfaces often hide important aspects of the search (by poor design or to protect proprietary relevance-ranking schemes), or make query specification so difficult and confusing that they discourage use. Evidence from empirical studies shows that users perform better and have higher subjective satisfaction when they can view and control the search (Koenemann and Belkin, 1996).

An analogy to the evolution of automobile user interfaces might clarify the goals. Early competitors offered a profusion of controls, and each manufacturer had a distinct design. Some designs—such as having a brake pedal that was far from the gas pedal—were dangerous. Furthermore, if you were accustomed to driving a car with the brake to the left of the gas pedal, and your neighbor's car had the reverse design, it might be risky to trade cars. It took a half-century to achieve good design and appropriate consistency in automobiles; let's hope that we can make the transition faster for text-search user interfaces.

Improved designs and consistency across multiple systems can bring faster performance, reduce mistaken assumptions, and increase success in finding relevant items. For example, with the variety of web search systems, such as Lycos, Infoseek, and AltaVista, users might expect that the search string `direct manipulation` would produce one of the following:

- Search on the exact string `direct manipulation`
- Probabilistic search for `direct` and `manipulation`
- Probabilistic search for `direct` and `manipulation`, with some weighting if the terms are in close proximity
- Boolean search on `direct` AND `manipulation`
- Boolean search on `direct` OR `manipulation`
- Error message indicating missing AND/OR operator or other delimiters

In many systems, there is little or no indication regarding which interpretation has been chosen and whether stemming, case matching, stop words, or other transformations are being applied. Often, the results are displayed in a relevance-ranked manner that is a mystery to many users (and sometimes is a proprietary secret).

To coordinate design practice, we might use a *four-phase framework* to satisfy the needs of first-time, intermittent, and frequent users who are accessing a variety of textual and multimedia libraries (Shneiderman et al., 1997). Finding common ground will be difficult; not finding it will be tragic. Although early adopters of technology are willing to overcome difficulties, the middle and late adopters are not so tolerant. The future of search services on the World Wide Web and elsewhere may depend on the degree to which user frustration and confusion are reduced, while the ability to find reliably sought items in the surging sea of information is increased.

The four-phase framework (Box 15.1) gives great freedom to designers to offer features in an orderly and consistent manner. The phases are

1. *Formulation:* expressing the search
2. *Initiation of action:* launching the search
3. *Review of results:* reading messages and outcomes
4. *Refinement:* formulating the next step

Formulation includes the *source* of the information, the *fields* for limiting the source, the *phrases*, and the *variants*. Even if technically and economically feasible, searching all libraries or collections in a library is not always the preferred approach. Users often prefer to limit the sources to a specific library, collection in a library, or subcollection range of items (users may choose date ranges, languages, media types, publishers, and so on). Users may wish to limit their search to specific *fields* (for example, the title, abstract, or full text of a scientific article) of items within a collection. Typically, users searching on common phrases would prefer to retrieve only those documents whose title contains those phrases. Sources may also be restricted by structured fields (year of publication, volume number, and so on).

In textual databases, users often seek items that contain meaningful *phrases* (`Civil War`, `Environmental Protection Agency`, `George Washington`, `air pollution`, `carbon monoxide`), and multiple entry windows should be provided to allow for multiple phrases. Searches on phrases have proved to be more accurate than are searches on words. Since some relevant items may be missed by a phrase approach, users should have the option to expand a search by breaking the phrases into separate words. Phrases also facilitate searching on names (for example, search on `George Washington` should not turn up George Bush or Washington, D.C.). If Boolean operations, proximity restrictions, or other combining strategies are specifiable, then the users should be able to express them. Users or service providers should have control over stop lists (common words, single letters, obscenities).

When users are unsure of the exact value of the field (subject term, or spelling or capitalization of a city name), they may want to relax the search constraints by allowing *variants* to be accepted. In structured databases, the variants may include a wider range on a numeric attribute. In a textual-document search, interfaces should allow user control over variant capitalization (case sensitivity), stemmed versions (the keyword `teach` retrieves variant suffixes such as teacher, teaching, or teaches), partial matches (the keyword `biology` retrieves sociobiology and astrobiology), phonetic variants from soundex methods (the keyword `Johnson` retrieves Jonson, Jansen, Johnsson), synonyms (the keyword `cancer` retrieves oncology), abbrevia-

Box 15.1

Four-phase framework to clarify user interfaces for textual search.

1. *Formulation*
   - Search the appropriate sources in libraries and collections
   - Use *fields* for limiting the source: structured fields such as year, media, or language; and text fields such as titles or abstracts of documents
   - Recognize *phrases* to allow entry of names such as `George Washington` or `Environmental Protection Agency`, and concepts such as `abortion rights reform` or `gallium arsenide`.
   - Permit *variants* to allow relaxation of search constraints, such as case sensitivity, stemming, partial matches, phonetic variations, abbreviations, or synonyms from a thesaurus.
2. *Action*
   - Include *explicit actions* initiated by buttons with consistent labels (such as "Search"), location, size, and color.
   - Include *implicit actions* initiated by changes to a parameter of the formulation phase that immediately produce a new set of search results.
3. *Results*
   - Read explanatory messages.
   - View textual lists.
   - Manipulate visualizations.
   - Control what the size of the result set is and which fields are displayed.
   - Change sequencing (alphabetical, chronological, relevance ranked, etc.).
   - Explore clustering (by attribute value, topics, etc.).
4. *Refinement*
   - Use meaningful messages to guide users in progressive refinement; for example, if the two words in a phrase are not found near each other, then offer easy selection of individual words or variants.
   - Make changing of search parameters convenient.
   - Allow search results and the setting of each parameter to be saved, sent by email, or used as input to other programs, such as visualization or statistical tools.

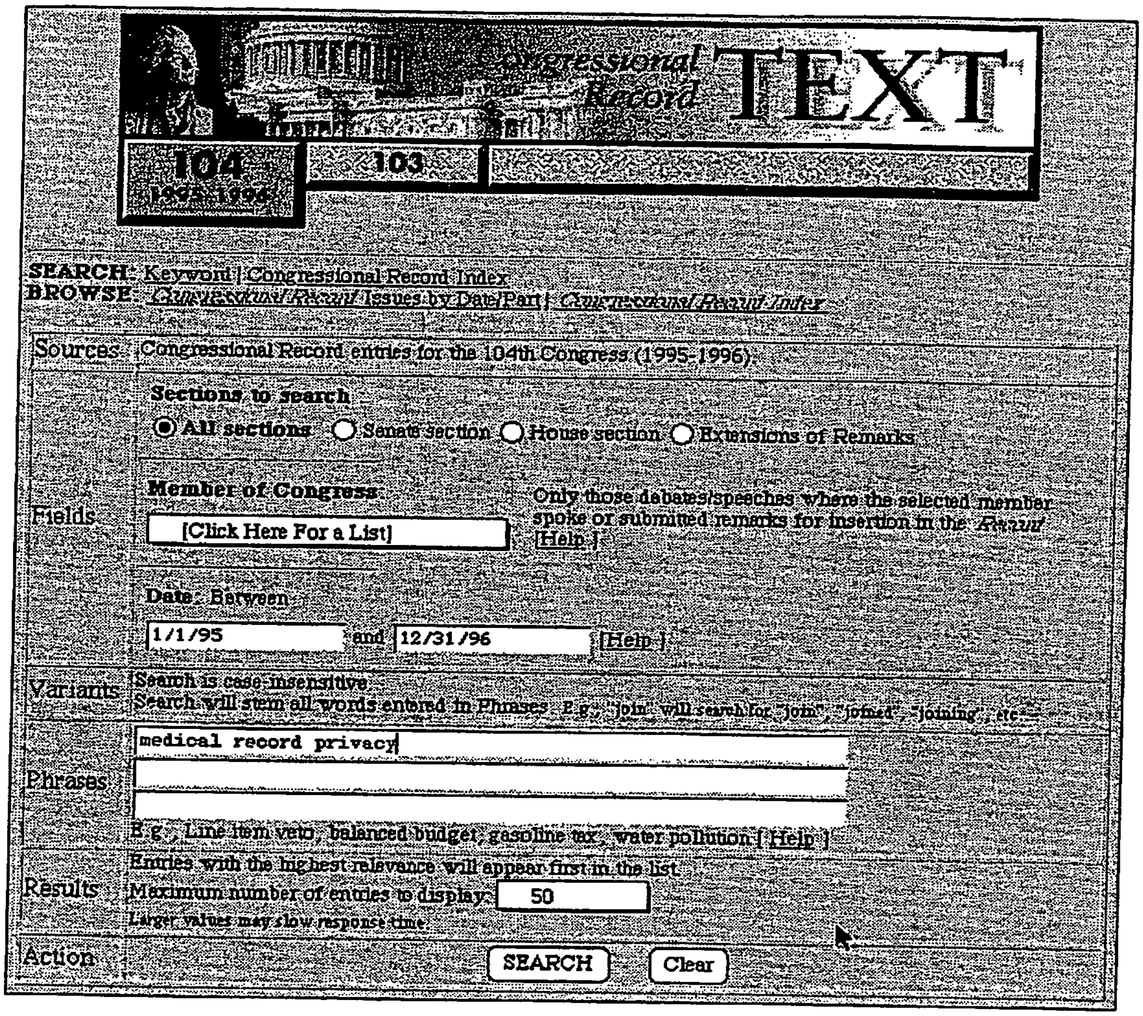
tions (the keyword `IBM` retrieves International Business Machines, and vice versa), and broader or narrower terms from a thesaurus (the keyphrase `New England` retrieves Vermont, Maine, Rhode Island, New Hampshire, Massachusetts, and Connecticut).

The second phase is the initiation of *action,* which may be explicit or implicit. Most current systems have a search button for explicit initiation, or for delayed or regularly scheduled initiation. The button label, size, and color should be consistent across versions. An appealing alternative is *implicit initiation,* in which each change to a component of the formulation phase immediately produces a new set of search results. *Dynamic queries*—in which users adjust query widgets to produce continuous updates—have proved to be effective and satisfying. They require adequate screen space and rapid processing, but their advantages are great.

The third phase is the review of *results,* in which the users read messages, view textual lists, or manipulate visualizations. Users may be given control over what the size of the result set is, which fields are displayed, how results are sequenced (alphabetical, chronological, relevance ranked), and how results are clustered (by attribute value, by topics) (Pirolli et al., 1996).

The fourth phase is the *refinement*. Search interfaces should provide meaningful messages to explain search outcomes and to support progressive refinement. For example, if a stop word, obscenity, or misspelling is eliminated from a search input window, or if stemmed terms, partial matches, or variant capitalizations are included, users should be able to see these changes to their query. If two words in a keyphrase are not found proximally, then feedback should be given about the occurrence of the words individually. If multiple phrases are input, then items containing all phrases should be shown first and identified, followed by items containing subsets; but if no documents are found with all phrases, that failure should be indicated. There is a fairly elaborate decision tree (maybe 60 to 100 branches) of search outcomes and messages that needs to be specified. Another aspect of feedback is that, as searches are made, the system should keep track of them in a *history buffer* to allow review of earlier searches. Progressive refinement, in which the results of a search are refined by changing of the search parameters, should be convenient. Search results and the settings of all parameters should be objects that can be saved, sent by electronic mail, or used as input to other programs—for example, for visualization or statistical tools.

The four-phase framework can be applied by designers to make the search process more visible, comprehensible, and controllable by users. This approach is in harmony with movement toward direct manipulation, in which the state of the system is made visible and is placed under user control. Novices may not want to see all the components of the four phases initially, but, if they are unhappy with the search results, they should be able to view the settings and change their queries easily. A revised interface for the Library of Congress' THOMAS system (Fig. 15.1) shows how the framework might be applied to full-text searching of proposed legislation.



SEARCH: Keyword | Congressional Record Index
BROWSE: *Congressional Record* Issues by Date/Part | *Congressional Record Index*

Sources: Congressional Record entries for the 104th Congress (1995-1996).

Fields:
Sections to search
[✓ All sections] Senate section House section Extensions of Remarks

Member of Congress: [Click Here For a List]
Only those debates/speeches where the selected member spoke or submitted remarks for insertion in the *Record* [Help]

Date: Between 1/1/95 and 12/31/96 [Help]

Variants: Search is case-insensitive.
Search will stem all words entered in Phrases. E.g., "join" will search for "join", "joined", "joining", etc.

Phrases: medical record privacy
E.g., Line item veto, balanced budget, gasoline tax, water pollution [Help]

Results: Entries with the highest relevance will appear first in the list.
Maximum number of entries to display: 50
Larger values may slow response time.

Action: SEARCH Clear

Figure 15.1

A revised interface for the Library of Congress' THOMAS system. The display shows how the four-phase framework might be applied to text searching on Congressional Record articles. Implemented by Bryan Slavin at the University of Maryland Human–Computer Interaction Laboratory. (Shneiderman et al., 1997.)

15.3 Multimedia Document Searches

Interfaces to search structured databases and textual-document libraries are good and getting better, but searching in multimedia document libraries is still in a primitive stage. Current approaches to locating images, videos, sound, or animation depend on a parallel database or document search to locate the items. For example, searches in photo libraries can be done by date, photographer, medium, location, or text in captions, but finding photos showing a ribbon-cutting ceremony or videos of a sunset is difficult. In the near term, those people who must search multimedia documents should push for ambitious captioning and attribute recording. Classification according to useful search categories (agriculture, music, sports, personalities) is helpful, although costly and imperfect.

Recent advances in computer algorithms may enable greater flexibility in locating items in multimedia libraries. User-interface designs to specify the permissible matches are varied. Some systems have elaborate textual commands, but most are moving toward graphical specification of query components:

- *Photo search* Finding photos with images such as the Statue of Liberty is a substantial challenge for image-analysis researchers, who describe this task as *query by image content (QBIC)*. Lady Liberty's distinctive profile might be identifiable if the orientation, lens focal length, and lighting were held constant, but the general problem is difficult in large and diverse collections of photos. Two promising approaches are to search for distinctive features such as the torch or the seven spikes in the crown, or to search for distinctive colors, such as the faded green copper verdigris. Users can specify features or color patterns with standard drawing tools, and even can indicate where in the image to search. For example, users could specify red, white, and blue in the upper third of an image to look for an American flag flying above a building. Of course, separating out the British, French, or other flags is not easy.

  More success is attainable with restricted collections, such as of glass vases, for which users could draw a desired profile and retrieve vases with long narrow necks. Other candidate collections include photos of constellations, subatomic particle tracks, or red blood cells. Users could specify their requests by selecting from a set of templates and adjusting the templates to describe their query. For critical applications, such as fingerprint matching, current successes depend on human identification of as many as 20 distinct features, but automatic recognition is improving. Even if completely automatic recognition is not possible, it will still be useful to have computers perform filtering, such as finding all the portraits with neutral backgrounds in a photo library.

- *Map search* Computer-generated maps are increasingly available online. Locating a map by latitude and longitude is the structured-database solution, but search by features is becoming possible because the tools used to build maps preserve the structural aspects and the multiple layers in maps. For example, users might specify a search for all port cities with a population greater than 1 million and an airport within 10 miles. Search on simpler maps such as airline routes might find flights to a given destination with no more than two connections on the same airline. Another candidate is weather maps, in which structured data—such as temperature, winds, or barometric pressure—make the search specification convenient.

- *Design or diagram search* Some computer-assisted design packages offer users limited search capabilities within a single design or across design collections. Finding red circles inside blue squares may help in some cases, but more elaborate strategies for finding engine designs with pistons smaller than 6 centimeters could prove more beneficial. Diagramming tools for making flowcharts or organization charts can add search capabilities to locate organizations that have more than five levels of management or situations where vice presidents are managing more than seven projects. Newspaper-layout packages could allow search for all occasions of headlines using fonts larger than 48 points, or headlines that span the front page.
- *Sound search* Imagine a music database system that would respond when users hum a few notes by producing a list of symphonies that contain that string of notes. Then, with a single touch, users could listen to the full symphony. Implementing this idea in the unstructured world of analog-encoded or even digitally encoded music is difficult, but imagine that the score sheets of symphonies were stored with the music and that string search over the score sheets was possible. Then, the application becomes easier to conceive. Identification of the users' hummed input might not be reliable, but if visual feedback were provided or if users entered the notes on a staff, then the fantasy would become feasible. Finding a spoken word or phrase in databases of telephone conversations is still difficult, but is becoming possible, even on a speaker-independent basis.
- *Video search* Searching a video or film involves more than simply searching through each of the frames. Users may wish to have a video segmented into scenes or cuts, and to identify zooming in or out and panning left or right. Gaining an overview of a 2-hour video by a time line of scenes would enable better understanding, editing, or selection. Combinations of structured databases and textual documents with video libraries lead to powerful services. Television news or sports libraries maintain structured databases and textual documents to support search for presidential appearances, disasters, or football highlights, carefully indexed for rapid future retrieval.
- *Animation search* Animation-authoring tools are still in early stages of development, but it might be possible to specify searches for certain kinds of animation—for example, spinning globes, moving banners, bouncing balls, or morphing faces. Although it might be less useful, it should be relatively easy to search for slides in a presentation that have moving text that comes in from the left, or in which the transition from one slide to another is by a barndoor animation.

15.4 Information Visualization

Grasping the whole is a gigantic theme. Arguably, intellectual history's most important. Ant-vision is humanity's usual fate; but seeing the whole is every thinking person's aspiration.

David Gelernter, *Mirror Worlds*, 1992

Visualization is a method of computing. It transforms the symbolic into the geometric, enabling researchers to observe their simulations and computations. Visualization offers a method for seeing the unseen. It enriches the process of scientific discovery and fosters profound and unexpected insights. In many fields it is already revolutionizing the way scientists do science.

McCormick et al., 1987

The success of direct-manipulation interfaces is indicative of the power of using computers in a more visual or graphic manner. A picture is often said to be worth a thousand words and, for some tasks, a visual presentation—such as a map or photograph—is dramatically easier to use or comprehend than is a textual description or a spoken report. As computer speeds and display resolution increase, information visualization and graphical interfaces are likely to have an expanding role. If a map of the United States is displayed, then it should be possible to point rapidly at one of 1000 cities to get tourist information. Of course, a foreigner who knows a city's name (for example, New Orleans), but does not know its location, may do better with a scrolling alphabetical list. Visual displays become even more attractive to provide orientation or context, to enable selection of regions, and to provide dynamic feedback for identifying changes (for example, on a weather map). Scientific visualization has the power to make visible and comprehensible atomic, cosmic, and common three-dimensional phenomena (such as heat conduction in engines, airflow over wings, or ozone holes). Abstract-information visualization has the power to reveal patterns, clusters, gaps, or outliers in statistical data, stock-market trades, computer directories, or document collections.

Overall, the bandwidth of information presentation is potentially higher in the visual domain than it is for media reaching any of the other senses. Humans have remarkable perceptual abilities that are greatly underutilized in current designs. Users can scan, recognize, and recall images rapidly, and can detect subtle changes in size, color, shape, movement, or texture. They can point to a single pixel, even in a megapixel display, and can drag one object to another to perform an action. User interfaces thus far have been largely text oriented, so as visual approaches are explored, appealing new opportunities are emerging.

There are many visual design guidelines. The central principle might be summarized as this *visual-information-seeking mantra:*

Overview first, zoom and filter, then details on demand
Overview first, zoom and filter, then details on demand
Overview first, zoom and filter, then details on demand
Overview first, zoom and filter, then details on demand
Overview first, zoom and filter, then details on demand
Overview first, zoom and filter, then details on demand
Overview first, zoom and filter, then details on demand
Overview first, zoom and filter, then details on demand
Overview first, zoom and filter, then details on demand
Overview first, zoom and filter, then details on demand
Overview first, zoom and filter, then details on demand
Overview first, zoom and filter, then details on demand

Each line represents one project in which I found myself rediscovering this principle and therefore wrote it down as a reminder. The mantra proved to be a good starting point when I was trying to characterize the multiple information-visualization innovations occurring at university, government, and industry research laboratories. To sort out the numerous prototypes and to guide researchers to new opportunities, Box 15.2 gives a *data type by task taxonomy (TTT)* of information visualizations.

As in the case of search, users are assumed to be viewing collections of items, where items have multiple attributes. In all seven data types (one-, two-, three-dimensional data; temporal and multi-dimensional data; and tree and network data) the items have one or more attributes. A basic search task is to select all items that match target attributes—for example, find all divisions in an company that have a budget greater than $500,000.

The data types of the TTT characterize the task-domain information objects and are organized by the problems that users are trying to solve. For example, in two-dimensional information such as maps, users are trying to grasp adjacency or to navigate paths, whereas in tree-structured information, users are trying to understand parent–child–sibling relationships. The tasks in the TTT are task-domain information actions that users wish to perform.

The seven tasks are at a high level of abstraction. Refinements and additions to these tasks would be natural next steps in expanding this taxonomy. The seven tasks are overview, zoom, filter, details-on-demand, relate, history, extract. Further discussion of the seven tasks follows the descriptions of the seven data types.

Box 15.2

Data Type by Task Taxonomy (TTT) to identify visualization data types and the tasks that need to be supported.

Data Type by Task Taxonomy (TTT)

Data Types

1-D Linear	Document Lens, SeeSoft Information Mural
2-D Map	GIS, ArcInfo, ThemeMap LyberWorld, InfoCrystal
3-D World	Desktops, WebBook, VRML CAD, Medical, Molecules
Temporal	Perspective Wall, ESDA MSProjects, LifeLines
Multi-Dimensional	Parallel Coordinates, Starfield, Visage Influence Explorer, TableLens
Tree	Outliners, Superbook, FileManager, Cone/Cam/Hyperbolic, TreeBrowser, Treemaps
Network	Netmap, SemNet, SeeNet, Butterfly

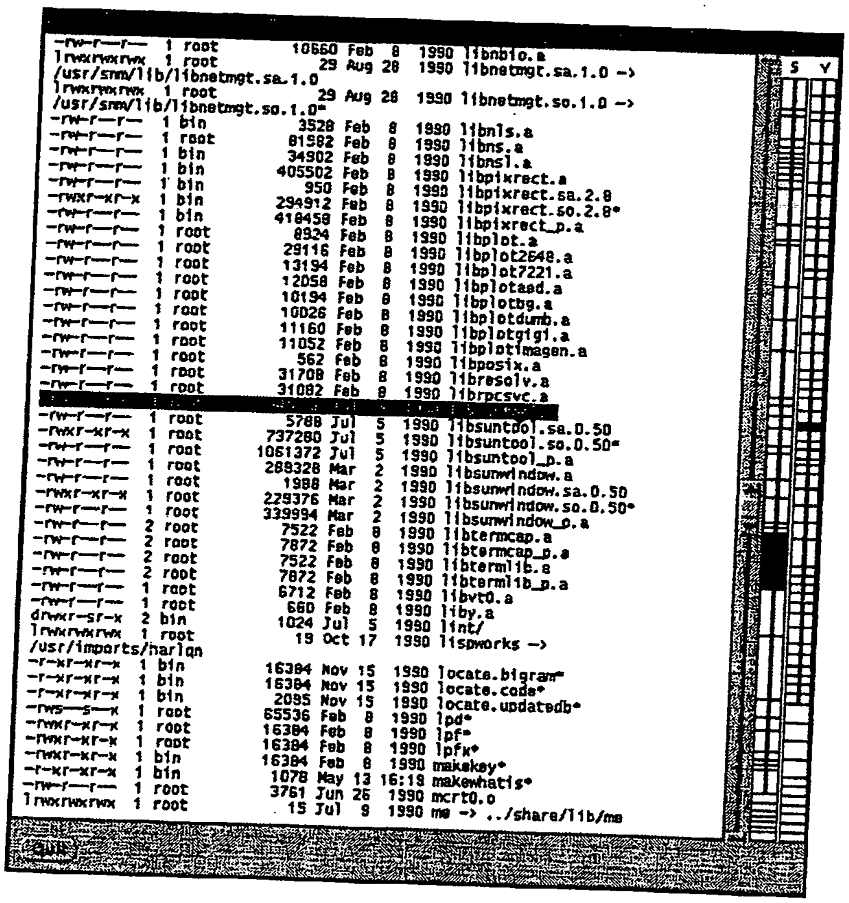
Tasks

Overview	Gain an overview of the entire collection.
Zoom	Zoom in on items of interest.
Filter	Filter out uninteresting items.
Details-on-demand	Select an item or group and get details when needed.
Relate	View relationships among items.
History	Keep a history of actions to support undo, replay, and progressive refinement.
Extract	Allow extraction of subcollections and of the query parameters.

1-D Linear Data

Linear data types include textual documents, program source code, and alphabetical lists of names, all of which are all organized in a sequential manner. Each item in the collection is a line of text containing a string of characters. Additional attributes might be the date of most recent update or author name. Interface-design issues include what fonts, color, size to use, and what overview, scrolling, or selection methods can be used. User tasks might be to find the number of items, to see items having certain attributes (show only lines of a document that are section titles, lines of a program that were changed from the previous version, or people in a list who are older than 21 years), or to see an item with all its attributes.

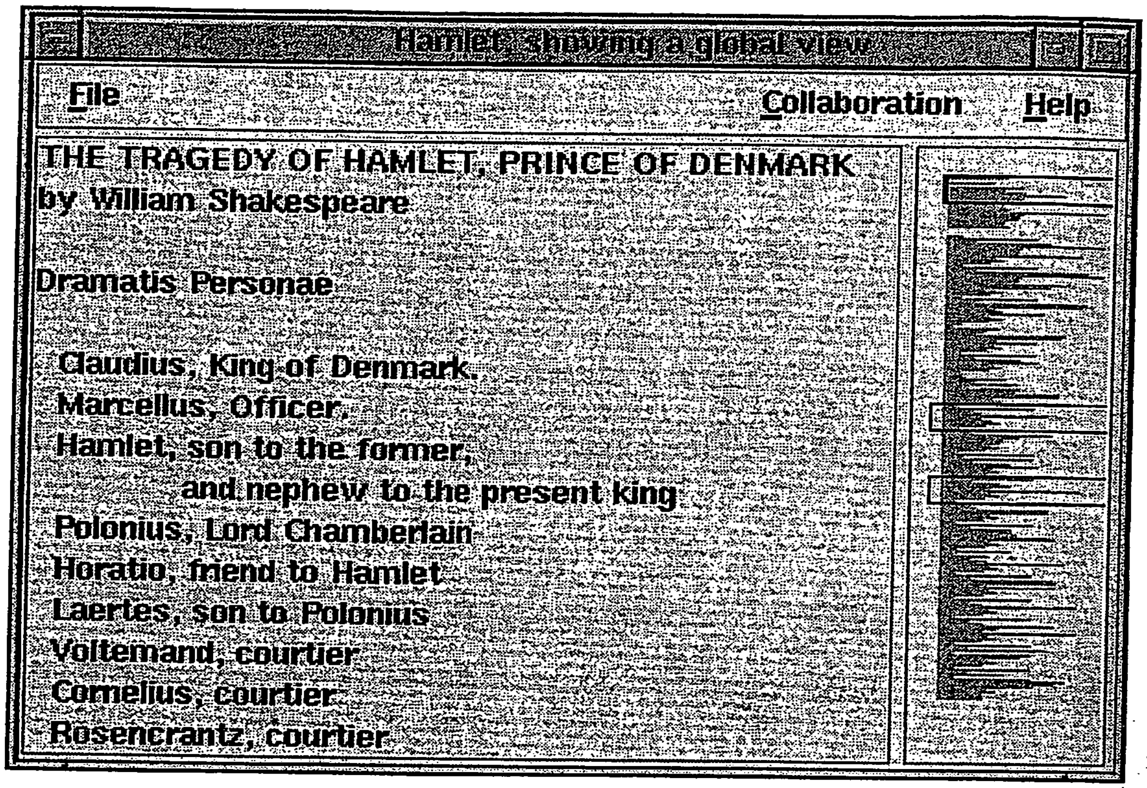
An early approach to dealing with large one-dimensional data sets was the *bifocal display,* which provided detailed information in the focus area and less information in the surrounding context area (Spence and Apperley, 1982). A selected issue of a scientific journal had details about each article; the older and newer issues of the journal were to the left and right on the bookshelf with decreasing space. Another effort to visualize one-dimensional data showed the attribute values of thousands of items in a fixed-sized space using a scrollbar-like display called *value bars* (Fig. 15.2) (Chimera, 1992). Even greater compressions were accomplished in compact displays of tens of thousands of lines of program source code in See Soft (Color Plate B1) (Eick et al., 1992) or lines in Hamlet (Fig. 15.3). Other examples of one-dimensional data include large textual documents in Document Lens (Fig. 15.4) (Robertson and Mackinlay, 1993) and historical


```
-rw-r--r--  1 root       10660 Feb  8  1990 libnbio.a
lrwxrwxrwx  1 root          29 Aug 28  1990 libnetmgt.sa.1.0 ->
/usr/snm/lib/libnetmgt.sa.1.0
lrwxrwxrwx  1 root          29 Aug 28  1990 libnetmgt.so.1.0 ->
/usr/snm/lib/libnetmgt.so.1.0*
-rw-r--r--  1 bin         3528 Feb  8  1990 libnls.a
-rw-r--r--  1 root       81582 Feb  8  1990 libns.a
-rw-r--r--  1 bin        34902 Feb  8  1990 libnsl.a
-rw-r--r--  1 bin       405502 Feb  8  1990 libpixrect.a
-rw-r--r--  1 bin          950 Feb  8  1990 libpixrect.sa.2.8
-rwxr-xr-x  1 bin       294912 Feb  8  1990 libpixrect.so.2.8*
-rw-r--r--  1 bin       418458 Feb  8  1990 libpixrect_p.a
-rw-r--r--  1 root        8924 Feb  8  1990 libplot.a
-rw-r--r--  1 root       29116 Feb  8  1990 libplot2648.a
-rw-r--r--  1 root       13194 Feb  8  1990 libplot7221.a
-rw-r--r--  1 root       12058 Feb  8  1990 libplotaed.a
-rw-r--r--  1 root       10194 Feb  8  1990 libplotbg.a
-rw-r--r--  1 root       10026 Feb  8  1990 libplotdumb.a
-rw-r--r--  1 root       11160 Feb  8  1990 libplotgigi.a
-rw-r--r--  1 root       11052 Feb  8  1990 libplotimagen.a
-rw-r--r--  1 root         562 Feb  8  1990 libposix.a
-rw-r--r--  1 root       31708 Feb  8  1990 libresolv.a
-rw-r--r--  1 root       31082 Feb  8  1990 librpcsvc.a

-rw-r--r--  1 root        5788 Jul  5  1990 libsuntool.sa.0.50
-rwxr-xr-x  1 root      737280 Jul  5  1990 libsuntool.so.0.50*
-rw-r--r--  1 root     1061372 Jul  5  1990 libsuntool_p.a
-rw-r--r--  1 root      289328 Mar  2  1990 libsunwindow.a
-rw-r--r--  1 root        1988 Mar  2  1990 libsunwindow.sa.0.50
-rwxr-xr-x  1 root      229376 Mar  2  1990 libsunwindow.so.0.50*
-rw-r--r--  1 root      339994 Mar  2  1990 libsunwindow_p.a
-rw-r--r--  2 root        7522 Feb  8  1990 libtermcap.a
-rw-r--r--  2 root        7872 Feb  8  1990 libtermcap_p.a
-rw-r--r--  2 root        7522 Feb  8  1990 libtermlib.a
-rw-r--r--  2 root        7872 Feb  8  1990 libtermlib_p.a
-rw-r--r--  1 root        6712 Feb  8  1990 libvt0.a
-rw-r--r--  1 root         660 Feb  8  1990 liby.a
drwxr-sr-x  2 bin         1024 Jul  5  1990 lint/
lrwxrwxrwx  1 root          19 Oct 17  1990 lispworks ->
/usr/imports/harlqn
-r-xr-xr-x  1 bin        16384 Nov 15  1990 locate.bigram*
-r-xr-xr-x  1 bin        16384 Nov 15  1990 locate.code*
-r-xr-xr-x  1 bin         2095 Nov 15  1990 locate.updatedb*
-rws--s--x  1 root       65536 Feb  8  1990 lpd*
-rwxr-xr-x  1 root       16384 Feb  8  1990 lpf*
-rwxr-xr-x  1 root       16384 Feb  8  1990 lpfx*
-rwxr-xr-x  1 bin        16384 Feb  8  1990 makekey*
-r-xr-xr-x  1 bin         1078 May 13 16:19 makewhatis*
-rw-r--r--  1 root        3761 Jun 26  1990 mcrt0.o
lrwxrwxrwx  1 root          15 Jul  9  1990 me -> ../share/lib/me
```

Figure 15.2

Each value bar shows one attribute of the linear list of items. In this Unix directory example, the two value bars on the right represent the file size (S) and file modification recency, or youth (Y). The currently selected file is one of the biggest in size and is moderately youthful. (Chimera, 1992.)



THE TRAGEDY OF HAMLET, PRINCE OF DENMARK
by William Shakespeare

Dramatis Personae

Claudius, King of Denmark.
Marcellus, Officer.
Hamlet, son to the former,
and nephew to the present king.
Polonius, Lord Chamberlain.
Horatio, friend to Hamlet.
Laertes, son to Polonius.
Voltemand, courtier.
Cornelius, courtier.
Rosencrantz, courtier.

Figure 15.3

Shakespeare's Hamlet, viewed with a one-dimensional overview on the right side showing where three users are reading the document. Each person's field-of-view box shows their location. This user is at the start. (Used with permission of the University of Calgary, Alberta, Canada.)

data about sunspots using the information-mural algorithms (Fig. 15.5) (Jerding and Stasko, 1995).

2-D Map data

Planar or map data include geographic maps, floorplans, and newspaper layouts. Each item in the collection covers some part of the total area and may or may not be rectangular. Each item has task-domain attributes, such as name, owner, and value, and interface-domain features, such as size, color, and opacity. Many systems adopt a multiple-layer approach to dealing with map data, but each layer is two-dimensional. User tasks are to find adjacent items, containing items and paths between items, and to perform the seven basic tasks.

Examples include geographic-information systems, which are a large research and commercial domain (Laurini and Thompson, 1992; Egenhofer and Richards, 1993) with numerous systems available (see Fig. 6.5). Information-visualization researchers have used spatial dis-

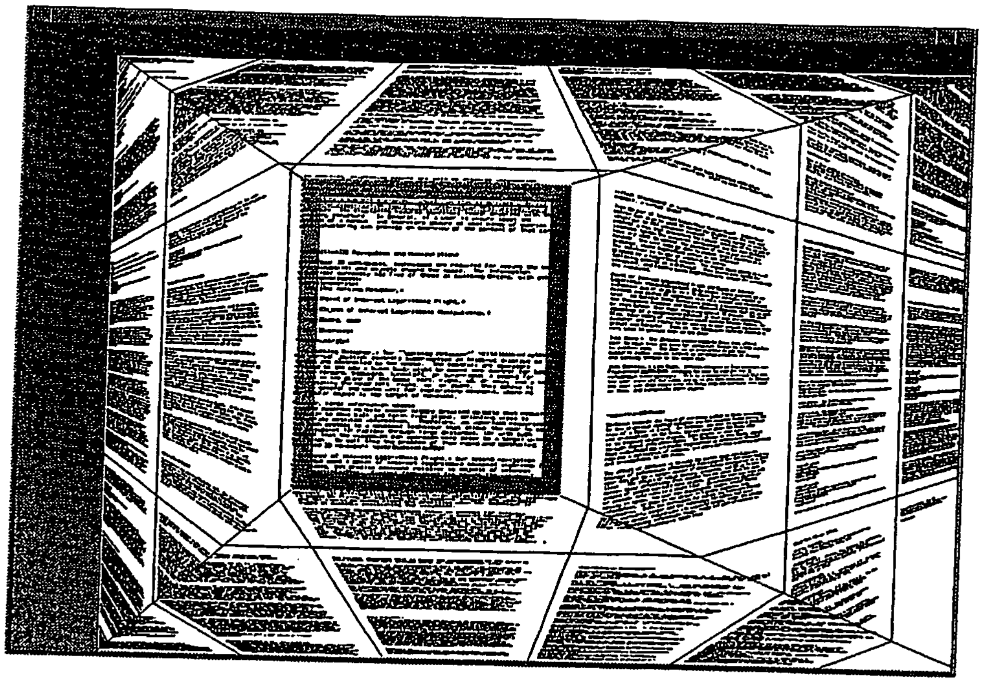


Figure 15.4

Document Lens showing many pages of a document in miniature form. Users can zoom in on any page easily and quickly. (Used with permission from Xerox PARC, Palo Alto, CA.)

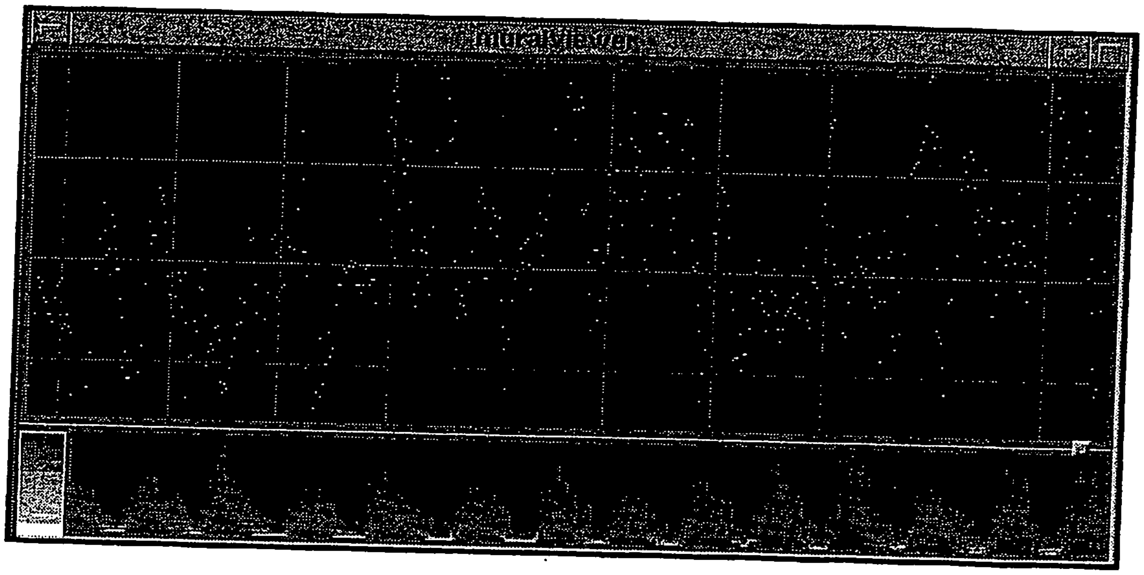


Figure 15.5

The information-mural overview at the bottom uses an antialiasing algorithm to show 52,000 readings of sun spots from 1850 to 1993. The field-of-view box at the bottom shows the context for the detail view on top. (Jerding & Stasko, 1995.) (Used with permission of Georgia Tech University, Atlanta, GA.)

plays of document collections (Color Plate B2) (Korfhage, 1991; Hemmje et al., 1993; Wise et al., 1995) organized proximally by term co-occurrences.

3-D World

Real-world objects such as molecules, the human body, and buildings have items with volume and with potentially complex relationships with other items. Computer-assisted design systems for architects, solid modelers, and mechanical engineers are built to handle complex three-dimensional relationships. Users' tasks deal with adjacency plus above–below and inside–outside relationships, as well as the seven basic tasks. In three-dimensional applications, users must cope with their position and orientation when viewing the objects, plus must handle the serious problems of *occlusion*. Solutions are proposed in many prototypes with techniques such as overviews, landmarks, perspective, stereo display, transparency, and color coding.

Examples of three-dimensional computer graphics and computer-assisted design are numerous, but information-visualization work in three dimensions is still novel. Some virtual-environment researchers have sought to present information in three-dimensional structures (see Section 6.8). Navigating high-resolution images of the human body is the challenge in the National Library of Medicine's Visible Human project (Fig. 15.6) (North et al., 1996). Architectural walkthroughs or flythroughs can give users an idea of what a finished building will look like. A three-dimensional desktop is thought to be appealing to users, but disorientation, navigation, and hidden data problems remain (Fig. 15.7) (Card et al., 1996).

Temporal data

Time lines are widely used and are sufficiently vital for medical records, project management, or historical presentations that researchers have created a data type that is separate from one-dimensional data. The distinctions of *temporal data* are that items have a start and finish time, and that items may overlap. Frequent tasks include finding all events before, after, or during some time period or moment, plus the seven basic tasks.

Many project-management tools exist; novel visualizations of time include the perspective wall (Fig. 15.8) (Robertson et al., 1993) and LifeLines (see Fig. 1.5 and Color Plate B3) (Plaisant et al., 1996). LifeLines shows a youth's history keyed to the needs of the Maryland Department of Juvenile Justice, but is intended to present medical patient histories as a compact overview with selectable items that allow users to get details-on-demand. Temporal-data visualizations appear in systems for editing video data, composing music, or preparing animations, such as Macromedia Director.

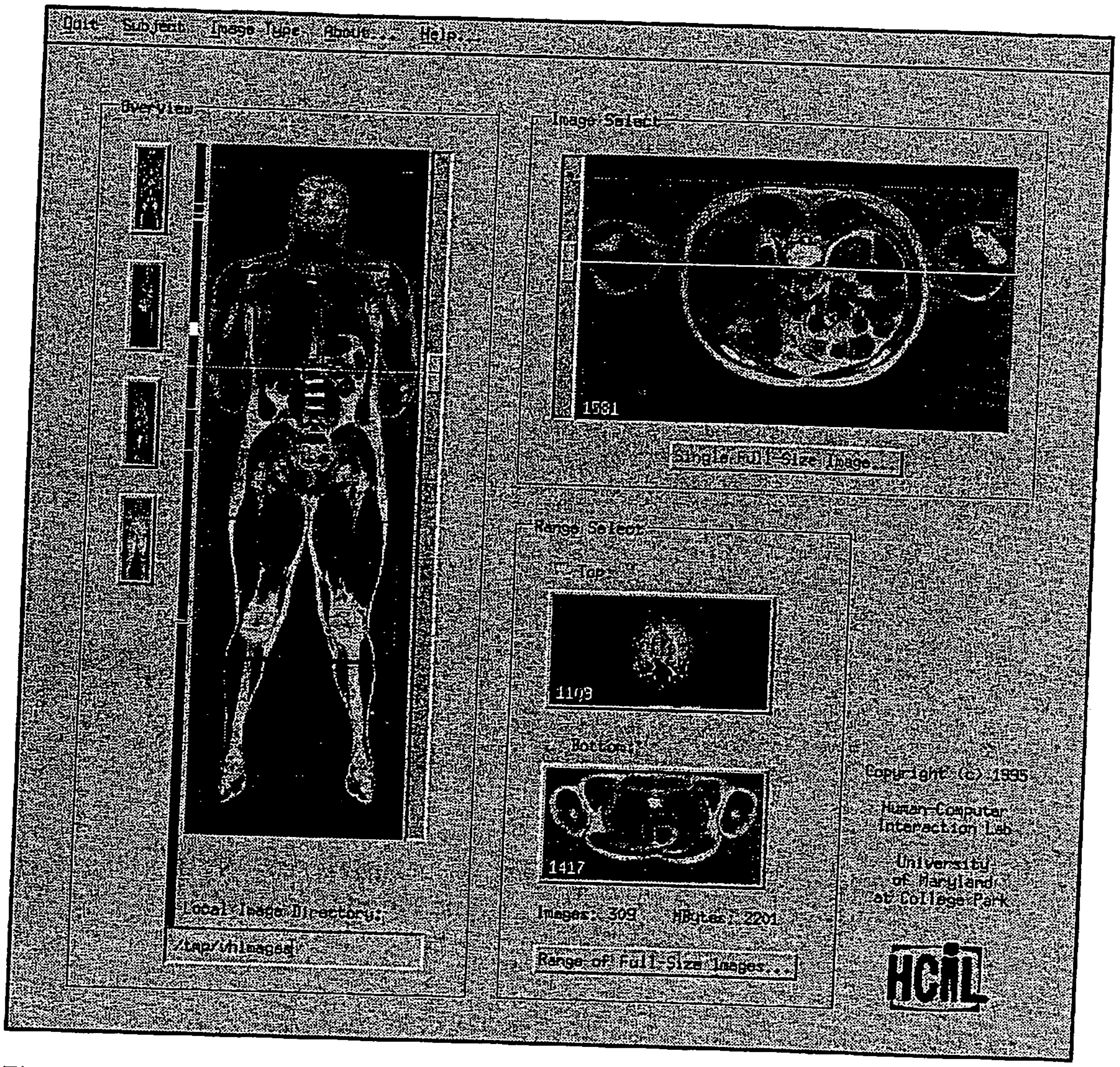


Figure 15.6

Visible Human Explorer user interface, showing a reconstructed coronal section overview (on the left) and an axial preview image of the upper abdominal region (on the upper right). Dragging the sliders animates the cross-sections through the body. (North et al., 1996.) (Available at http://www.nlm.nih.gov)

Multidimensional data

Most relational- and statistical-database contents are conveniently manipulated as multidimensional data, in which items with *n* attributes become points in a *n-dimensional space*. The interface representation can be dynamic two-dimensional scattergrams, with each additional dimension controlled by a slider (Ahlberg and Shneiderman, 1994). Buttons can used

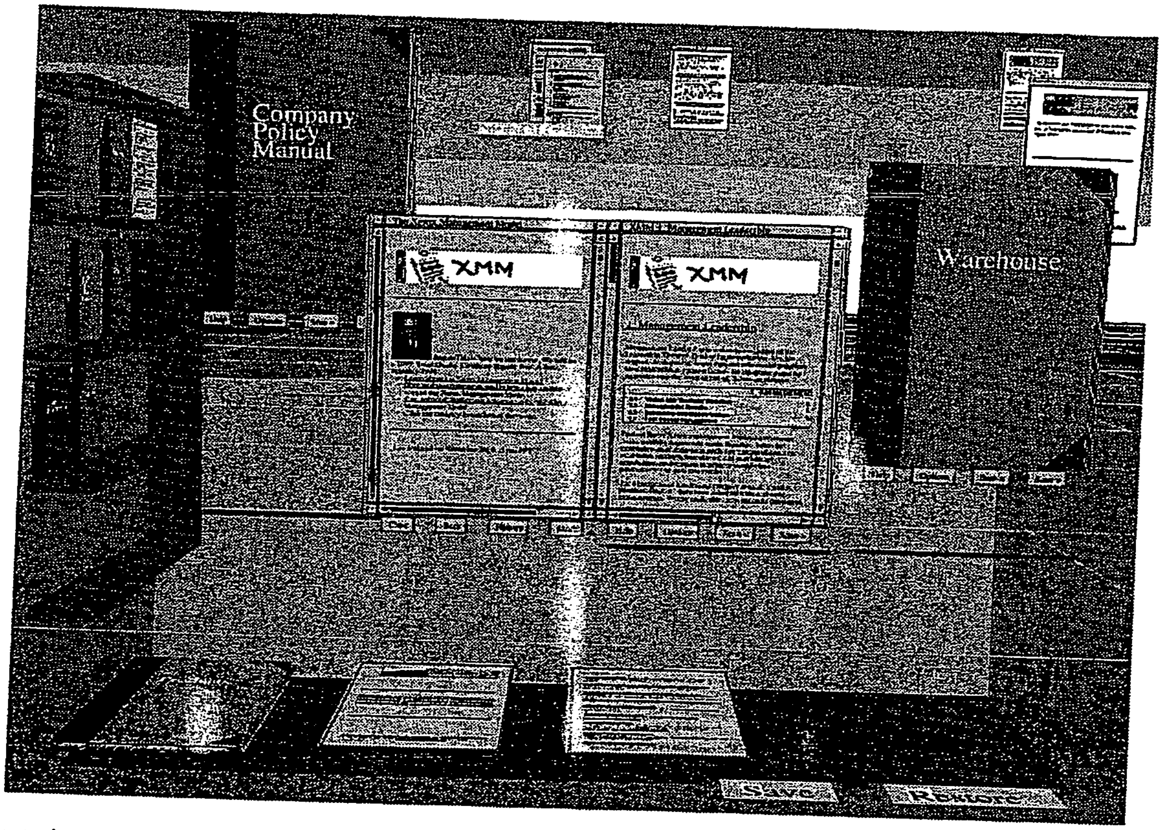


Figure 15.7

WebBook and WebForager. These three-dimensional worlds are used for browsing and recording web pages. (Used with permission from Xerox PARC, Palo Alto, CA.)

for attribute values when the cardinality is small—say, less than 10. Tasks include finding patterns, clusters, correlations among pairs of variables, gaps, and outliers. Multidimensional data can be represented by a three-dimensional scattergram, but disorientation (especially if the user's point of view is from inside the cluster of points) and occlusion (especially if close points are represented as being larger) can be problems. The technique of using parallel coordinates (Fig. 15.9) is a clever innovation that makes certain tasks easier, but takes practice for users to comprehend (Inselberg, 1985).

The early HomeFinder (Fig. 15.10) developed dynamic queries and sliders for user-controlled visualization of multidimensional data (Williamson and Shneiderman, 1992). The successor FilmFinder (Color Plate B4a–c) refined the techniques (Ahlberg and Shneiderman, 1994) for starfield displays (zoomable, color-coded, user-controlled scattergrams), and laid the basis for the commercial product Spotfire (Color Plate B5) (Ahlberg and Wistrand, 1995). Extrapolations include the Aggregate Manipulator (Goldstein and Roth, 1994), movable filters (Fishkin and

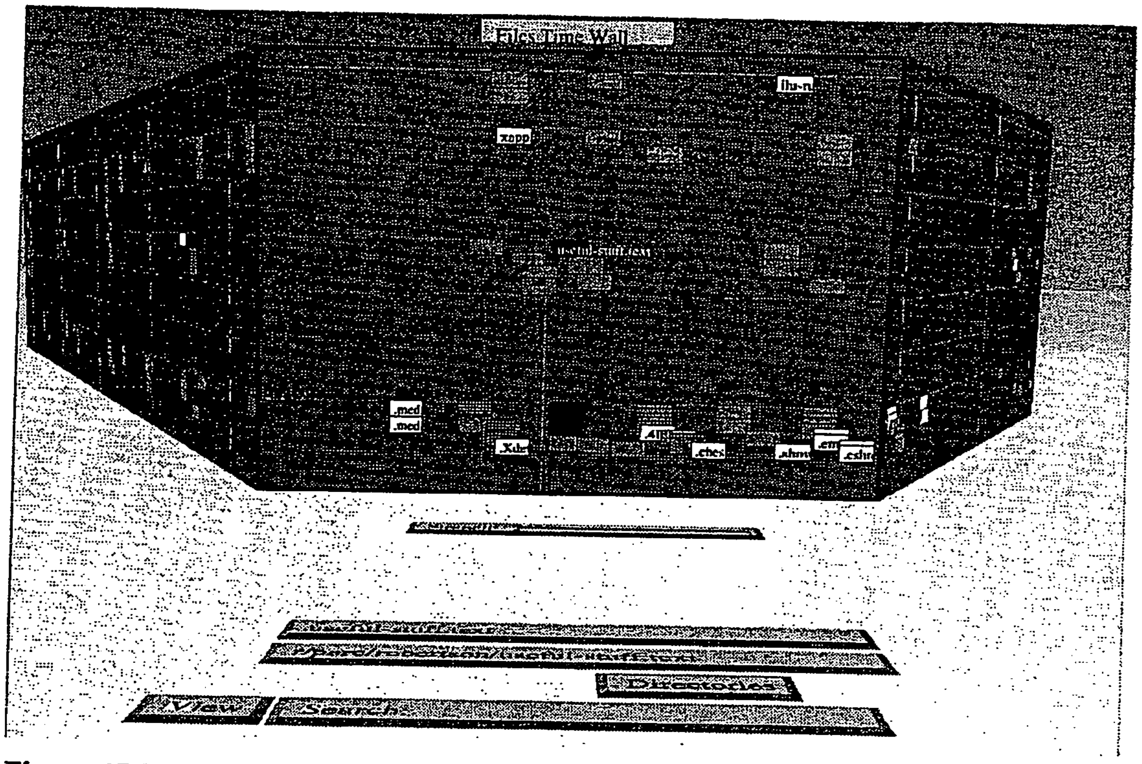


Figure 15.8

A perspective wall, showing time moving from left to right, with the focus in the center. Different categories of programs are shown on each level of the wall. Color or size coding can be used. (Used with permission from Xerox PARC, Palo Alto, CA.)

Stone, 1995), and Selective Dynamic Manipulation (Chuah et al., 1995). Related works include VisDB for multidimensional database visualization (Keim and Kreigal, 1994), the spreadsheet-like Table Lens (Fig. 15.11) (Rao and Card, 1994), and the multiple linked histograms in the Influence Explorer (Tweedie et al., 1996).

Tree data

Hierarchies or tree structures are collections of items, in which each item (except the root) has a link to one parent item. Items and the links between parent and child can have multiple attributes. The basic tasks can be applied to items and links, and tasks related to structural properties become interesting—for example, how many levels are in the tree, or how many children does an item have? While it is possible to have similar items at leaves and internal nodes, it is also common to find different items at each level in a tree. Fixed-level trees, with all leaves equidistant from the root, and fixed-fanout trees, with the same number of children for every parent, are easier to handle. High-fanout (broad) and small-fanout (deep) trees are important special cases. Inter-

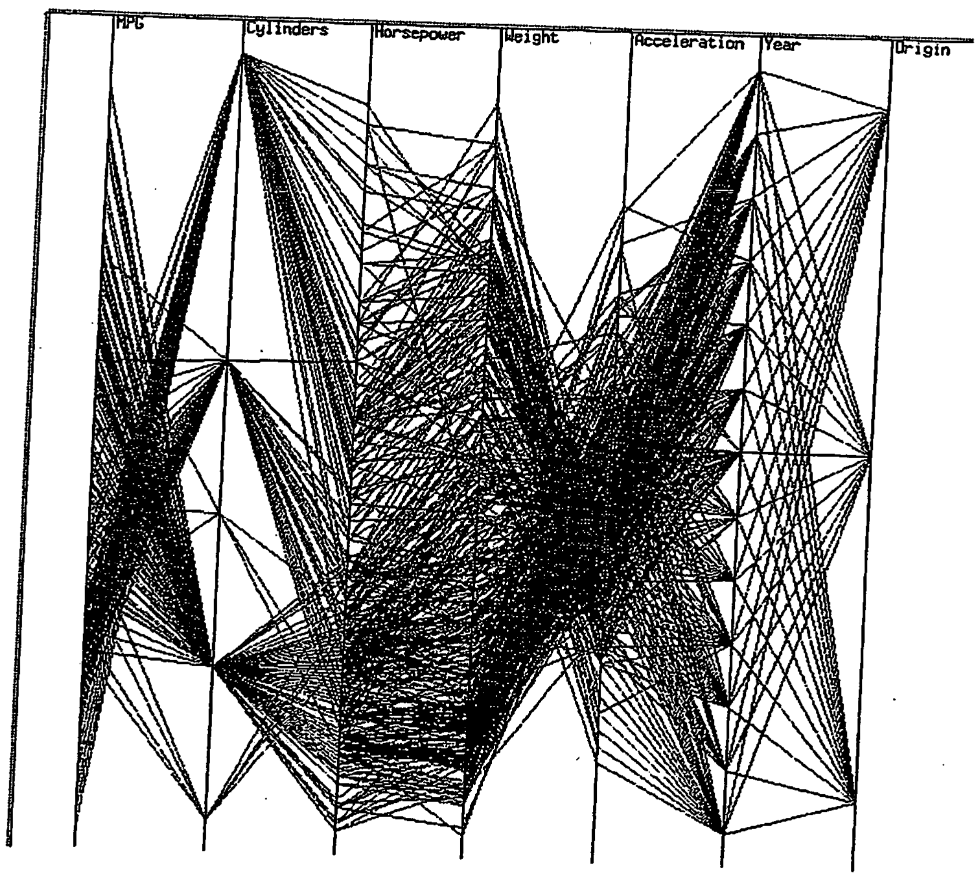
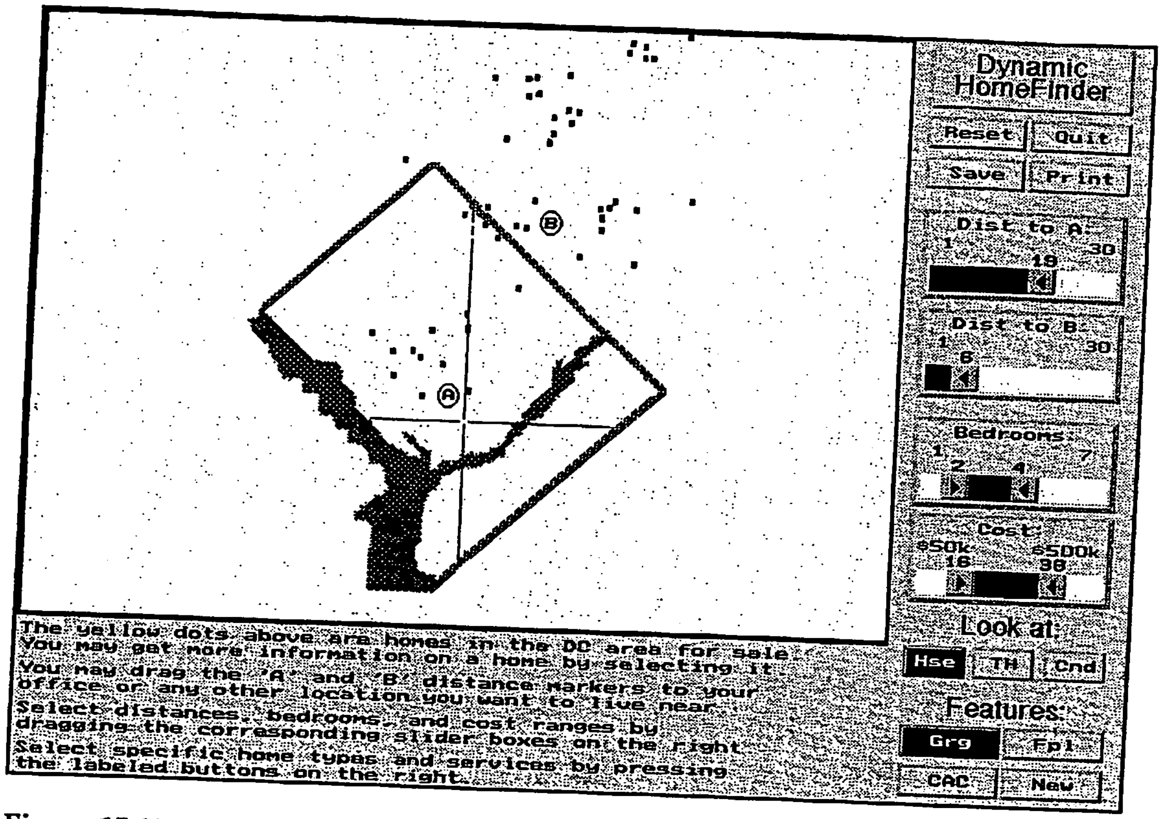


Figure 15.9

Parallel coordinate plot of seven dimensions of automobile data (CARS dataset obtained from StatLib at Carnegie Mellon University). There is a range of MPG (miles per gallon) values, but clear clusters of two-, four-, and six-cylinder cars are visible. More cylinders generally produce higher horsepower. Also notable is the generally inverse relationship of weight to acceleration. (Used with permission of Matt Ward, Worcester Polytechnic Institute, Worcester, MA.)

face representations of trees can use the outline style of indented labels used in tables of contents (Chimera and Shneiderman, 1993); a node-and-link diagram; or a *treemap*, in which child items are rectangles nested inside parent rectangles.

Tree-structured data has long been displayed with indented outlines (Egan et al., 1989), or with connecting lines, as in many computer-directory file managers. Attempts to show large tree structures as node-and-link diagrams in compact forms include the three-dimensional cone (Fig. 15.12) and cam trees (Robertson et al., 1993; Carriere and Kazman, 1995), dynamic pruning in the TreeBrowser (Fig. 15.13) (Kumar et al., 1997), and the appealingly animated hyperbolic trees (Fig. 15.14) (Lamping et al.,



The yellow dots above are homes in the DC area for sale.
You may get more information on a home by selecting it.
You may drag the 'A' and 'B' distance markers to your
office or any other location you want to live near.
Select distances, bedrooms, and cost ranges by
dragging the corresponding slider boxes on the right.
Select specific home types and services by pressing
the labeled buttons on the right.

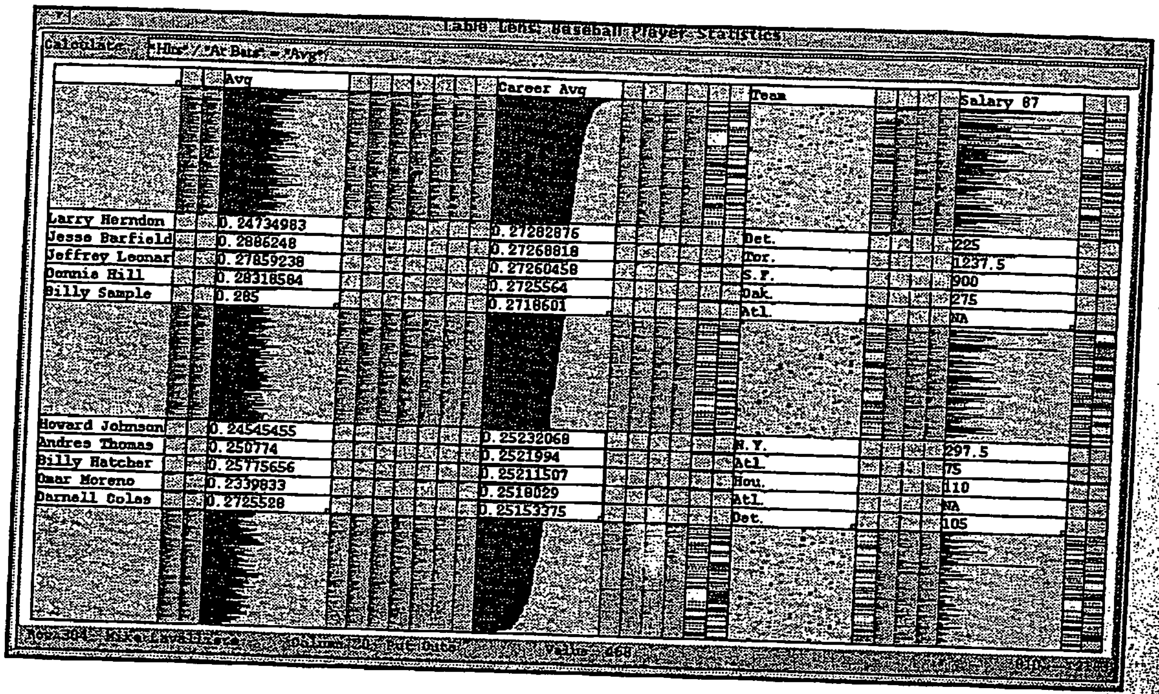
Figure 15.10

Dynamic HomeFinder, an early application of dynamic queries. Homes for sale in the Washington, D.C., area were shown as 1100 points of light. As users moved the sliders on the right, the screen was updated immediately to show the points matching the current query. By clicking on any point, users could get a detailed description. (Williamson and Shneiderman, 1992.)

1995). The space-filling mosaic approach, treemaps, shows an arbitrary-sized tree in a fixed rectangular space (Shneiderman, 1992; Johnson and Shneiderman, 1991). The treemap approach was applied successfully to libraries (Fig. 15.15), computer directories (Fig. 15.16), sales data, business decision making (Asahi et al., 1995), and web browsing (Mitchell et al., 1995; Mukherjea et al., 1995), but first-time users take 10 to 20 minutes to accommodate to treemaps.

Network data

Sometimes, relationships among items cannot be captured conveniently with a tree structure, and it is useful to have items linked to an arbitrary number of other items. Although many special cases of networks exist (acyclic, lattices, rooted versus unrooted, directed versus undirected), it is



	Avg	Career Avg	Team	Salary 87
Larry Herndon	0.24734983	0.27282876	Det.	225
Jesse Barfield	0.2886248	0.27268818	Tor.	1237.5
Jeffrey Leonar	0.27859238	0.27260458	S.F.	900
Donnie Hill	0.28318584	0.2725564	Oak.	275
Billy Sample	0.285	0.2718601	Atl.	NA
Howard Johnson	0.24545455	0.25232068	N.Y.	297.5
Andres Thomas	0.250774	0.2521994	Atl.	75
Billy Hatcher	0.25775656	0.25211507	Hou.	110
Omar Moreno	0.2339833	0.2518029	Atl.	NA
Darnell Coles	0.2725528	0.25153375	Det.	105

Figure 15.11

Table Lens, a program that provided a spreadsheetlike world that also supported information-visualization methods to find rankings and correlations among baseball players. (Used with permission from Xerox PARC, Palo Alto, CA.)

convenient to consider them all as one data type. In addition to performing the basic tasks applied to items and links, network users often want to know about shortest or least costly paths connecting two items or traversing the entire network. Interface representations include a node-and-link diagram, and a square matrix of the items with the value of a link attribute in the row and column representing a link.

Network visualization is an old but still imperfect art because of the complexity of relationships and user tasks. Commercial packages can handle small networks or simple strategies, such as Netmap's layout of nodes on a circle with links criss-crossing the central area. Specialized visualizations can be designed to be more effective for a given task, such as a network diagram showing heavy telephone traffic on holidays (Color Plate B6). An ambitious three-dimensional approach allowed users to fly into a network and control the visualization (Fairchild et al., 1988). New interest in this topic has been spawned by attempts to visualize the World Wide Web (Andrews, 1995; Hendley et al., 1995).

The seven data types that we have discussed reflect an abstraction of the reality. There are many variations on these themes (two-and-one-half

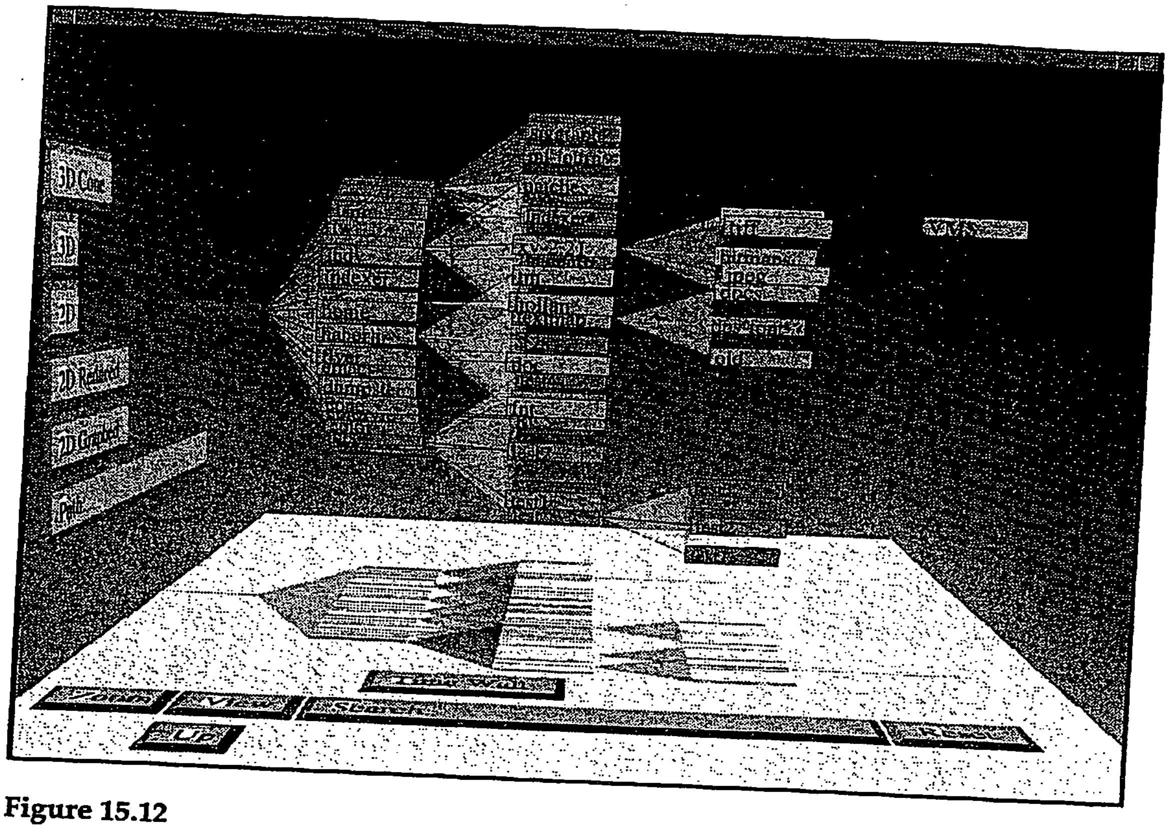


Figure 15.12

Cam-tree design, showing a hierarchical directory from left to right side. Users can rotate the trees smoothly in this three-dimensional viewer. When the root of the tree is shown at the top, this representation is called a cone-tree. (Used with permission from Xerox PARC, Palo Alto, CA.)

or four-dimensional data, multitrees) and many prototypes use combinations of these data types. This taxonomy is useful only if it facilitates discussion and leads to useful discoveries. We can get an idea of missed opportunities by looking at the tasks and data types in depth.

Overview task

We can gain an overview of the entire collection. Overview strategies (Section 13.5) include zoomed-out views of each data type that allow the user to see the entire collection plus an adjoining detail view. The overview contains a movable field-of-view box with which the user controls the contents of the detail view, allowing zoom factors of 3 to 30. Replication of this strategy with intermediate views enables users to reach larger zoom factors. Another popular approach is the fisheye strategy (Furnas, 1986), which has been applied most commonly for network browsing (Fig.

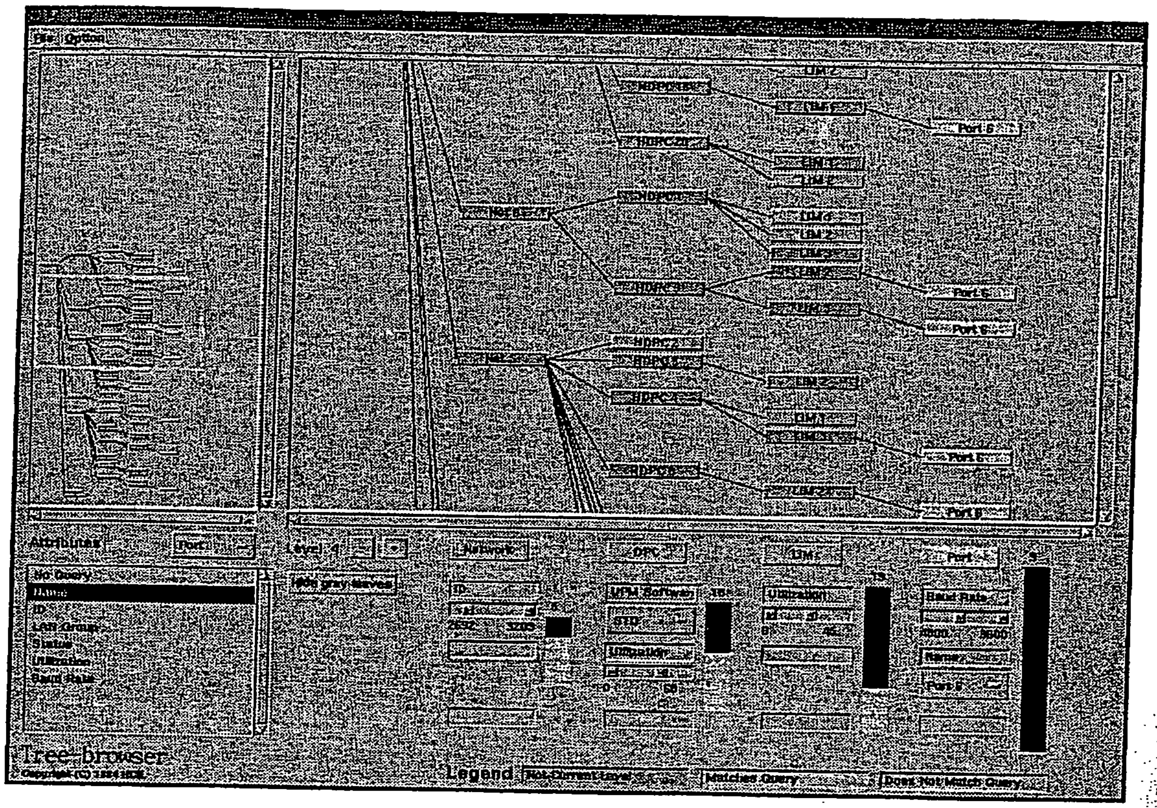


Figure 15.13

PDQ TreeBrowser, which supports pruning of nodes at every level of a tree. A user has pruned an 1100-node tree of a satellite network, using dynamic query sliders at four levels; only nine possible ports (leaf nodes) remain in the result set. (Kumar et al., 1997)

15.17) (Sarkar and Brown, 1994; Bartram et al., 1995; Schaffer et al., 1996). The fisheye distortion magnifies one or more areas of the display, but zoom factors in prototypes are limited to about 5. Although query-language facilities made it difficult to gain an overview of a collection, information-visualization interfaces support some overview strategy—or should do so. Adequate overview strategies are a useful criterion to judge such interfaces. In addition, look for navigation tools to pan or scroll through the collection.

Zoom task

We can zoom in on items of interest. Users typically have an interest in some portion of a collection, and they need tools to enable them to control the zoom focus and the zoom factor. Smooth zooming helps users to pre-

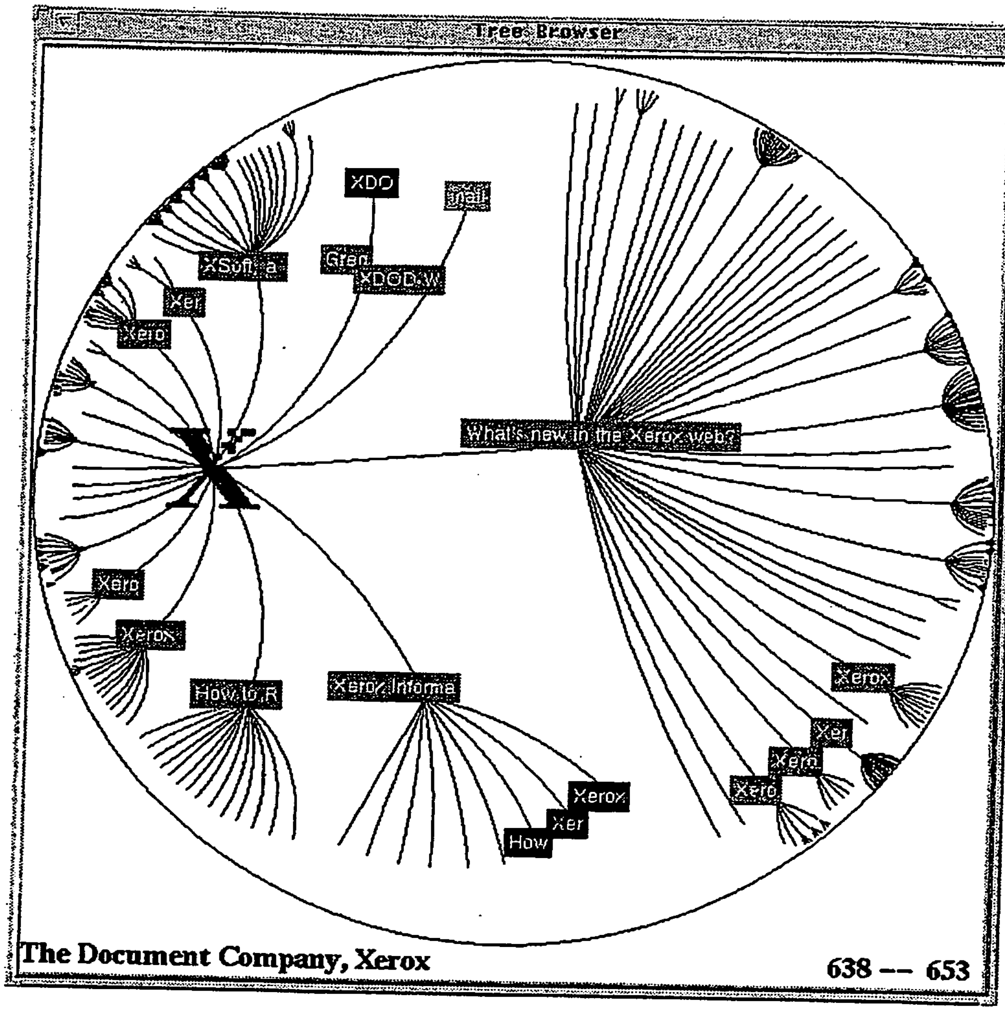
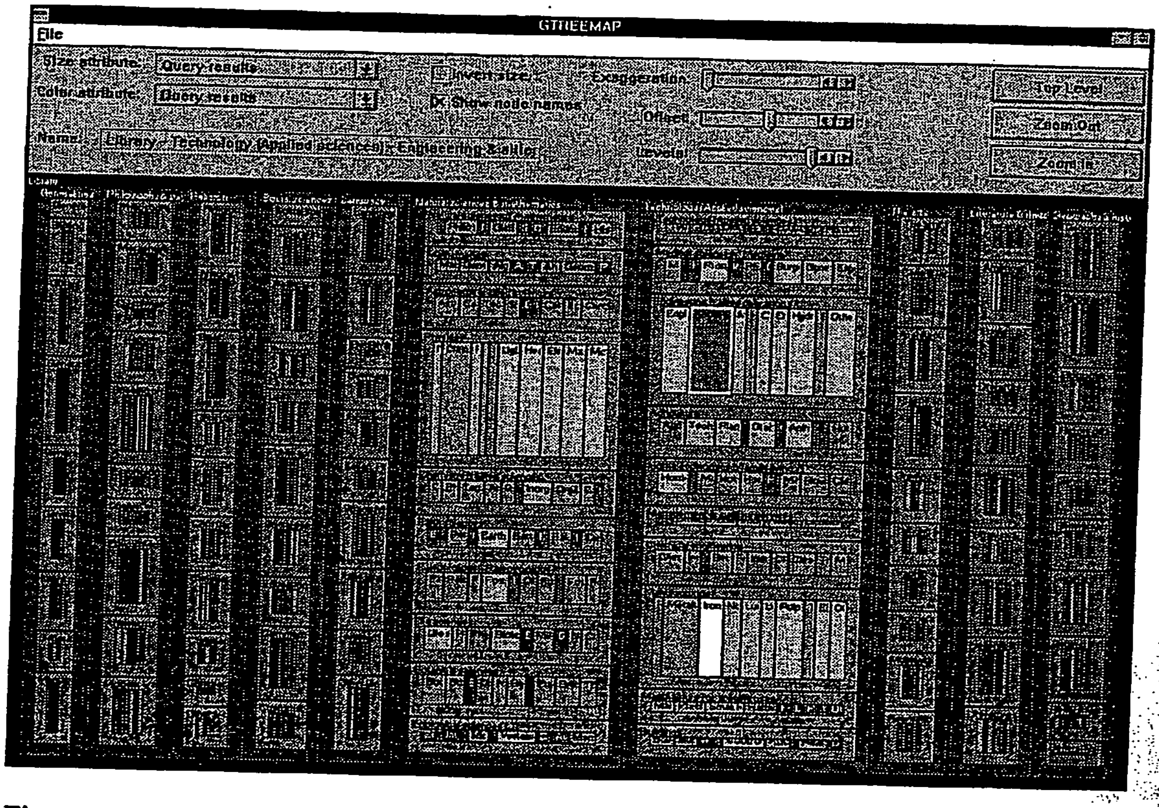


Figure 15.14

A hyperbolic tree browser that allows 10 to 30 nodes near the center to be seen clearly; branches are reduced gradually as they get closer to the periphery. This display technique guarantees that large trees can be accommodated in a fixed screen size. As the focus is shifted among nodes, the display updates smoothly, producing a satisfying animation. Landmarks or other features can be introduced to reduce the disorienting effect of movement (Lamping, Rao et al., 1995). (Used with permission of InXight Software, Palo Alto, CA.)

serve their sense of position and context (Schaffer et al., 1996). A user can zoom on one dimension at a time by moving the zoombar controls or by adjusting the size of the field-of-view box. A satisfying way to zoom in is to point to a location and to issue a zooming command, usually by holding down a mouse button (Bederson and Hollan, 1993). Zooming in one dimension has proved useful in starfield displays (Jog and Shneiderman, 1995).



Size attribute: Query results

Color attribute: Query results

Invert size

[X] Show node names

Exaggeration

Offset

Levels

Name: Library / Technology (Applied sciences) / Engineering & allied

Figure 15.15

The first three levels of the Dewey Decimal System shown as a treemap in which size indicates the number of books held in each of the 1000 categories. Color indicates frequency of utilization, with darker indicating high utilization (hot) and lighter indicating low utilization. Implemented by Marko Teittinen at the University of Maryland Human–Computer Interaction Laboratory.

Filter task

We can filter out uninteresting items. Dynamic queries applied to the items in the collection constitute one of the key ideas in information visualization (Ahlberg et al., 1992; Williamson and Shneiderman, 1992; Kumar et al., 1997). When users control the contents of the display, they can quickly focus on their interests by eliminating unwanted items. Sliders, buttons, or other control widgets coupled to rapid (less than 100 milliseconds) display update is the goal, even when there are tens of thousands of displayed items.

Details-on-demand task

We can select an item or group to get details. Once a collection has been trimmed to a few dozen items, it should be easy to browse the details about the group or individual items. The usual approach is to simply click on an item to get a pop-up window with values of each of the attributes.

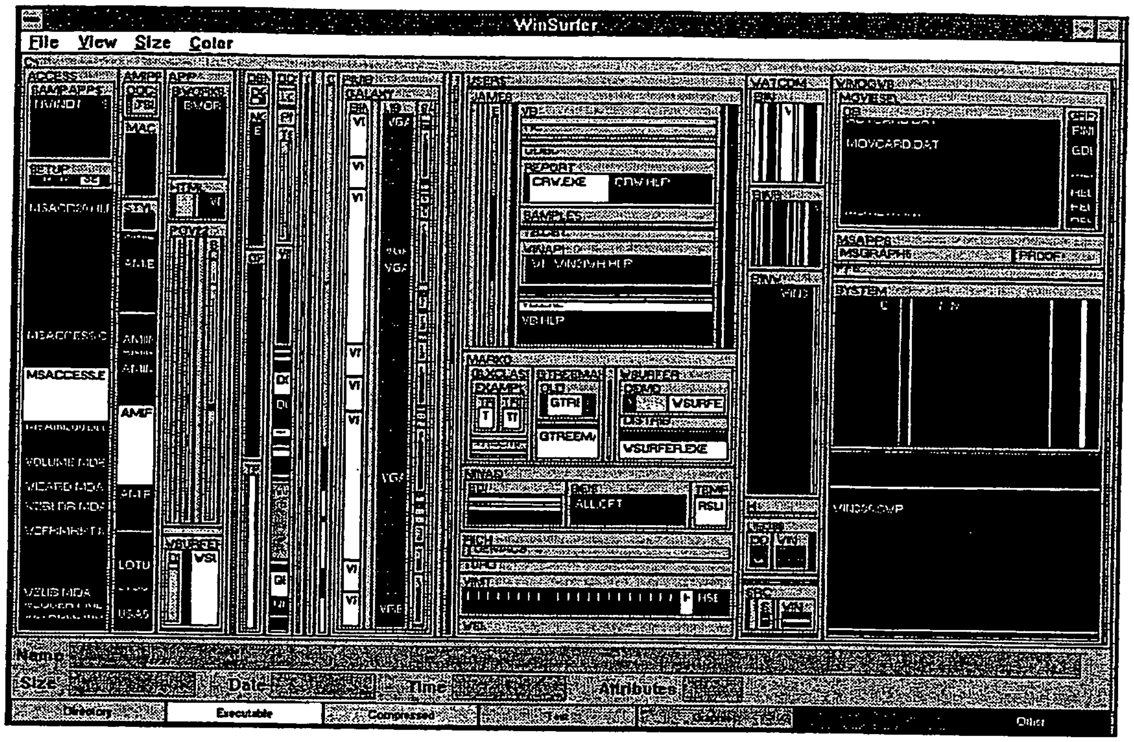


Figure 15.16

Winsurfer treemap that shows the 4900 files at several levels on a hard disk. Area is set to be proportional to file size and color to file type. Moving the cursor over an area produces an immediate display of attribute values on the bottom. Developed by Marko Teittinen at the University of Maryland Human–Computer Interaction Laboratory.

In Spotfire (Color Plate B5), the details-on-demand window can contain HTML text with links to further information.

Relate task

We can view relationships among items. In the FilmFinder details-on-demand window (Ahlberg and Shneiderman, 1994), users could select an attribute, such as the film's director, and cause the director alphaslider to be reset to the director's name, thereby displaying only films by that director. Similarly, in SDM (Chuah et al., 1995), users can select an item and then highlight items with similar attributes. In LifeLines (Color Plate B3) (Plaisant et al., 1996), users can click on a medication and see the related visit report, prescriptions, and laboratory test results. Designing user-interface actions to specify which relationship is to be manifested is still a challenge. The Influence Explorer (Tweedie et al., 1996) emphasizes exploration of relationships among attributes. The Table Lens (Fig. 15.11) emphasizes finding correlations among pairs of numerical attributes (Rao and Card, 1994).

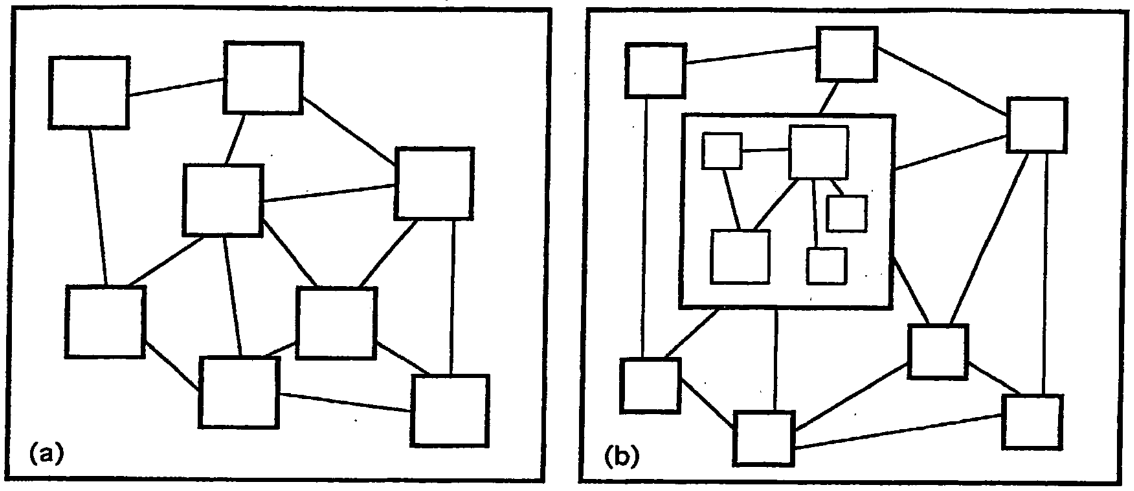


Figure 15.17

A fisheye view or variable zooming on a hierarchical network diagram. These techniques can help to focus attention on details while preserving context. (a) A central node has been selected for zooming. (b) The node is expanded by a zoom factor of 3, exposing five nodes at the next level. (Schaffer et al., 1996.)

History task

We can keep a history of actions to support undo, replay, and progressive refinement. It is rare that a single user action produces the desired outcome. Information exploration is inherently a process with many steps, so keeping the history of actions and allowing users to retrace their steps is important. However, most prototypes fail to deal with this requirement. Maybe they are reflecting the current state of GUIs, but designers would do better to model information-retrieval systems, which typically preserve the sequence of searches so that these searches can be combined or refined.

Extract task

We can allow extraction of subcollections and of the query parameters. Once users have obtained the item or set of items that they desire, it would be useful for them to be able to extract that set and to save it to a file in a format that would facilitate other uses, such as sending by electronic mail, printing, graphing, or insertion into a statistical or presentation package. As an alternative to saving the set, they might want to save, send, or print the settings for the control widgets. Few prototypes support such actions, although Roth's recent work on Visage provides an elegant capability to extract sets of items and simply drag-and-drop them into the next application window (Roth et al., 1996).

The attraction of visual displays, when compared to textual displays, is that they make use of the remarkable human perceptual ability for visual information. Within visual displays, there are opportunities for showing relationships by proximity, by containment, by connected lines, or by color coding. Highlighting techniques (for example, boldface text or brightening, inverse video, blinking, underscoring, or boxing) can be used to draw attention to certain items in a field of thousands of items. Pointing to a visual display can allow rapid selection, and feedback is apparent. The eye, the hand, and the mind seem to work smoothly and rapidly as users perform actions on visual displays.

15.5 Advanced Filtering

Users have highly varied needs for filtering features. The dynamic-queries approach of adjusting numeric range sliders, alphasliders for names or categories, or buttons for small sets of categories is appealing to many users for many tasks (Shneiderman, 1994). Dynamic queries might be called *direct-manipulation queries*, since they share the same concepts of visual display of actions (the sliders or buttons) and objects (the query results in the task-domain display); the use of rapid, incremental, and reversible actions; and the immediate display of feedback (less than 100 milliseconds). Additional benefits are the prevention of syntax errors and an encouragement of exploration.

Dynamic queries can reveal global properties, as well as assist users in answering specific questions. As the database grows, it is more difficult to update the display fast enough, and specialized data structures or parallel computation is required. Dynamic queries have attracted attention, although many user-interface problems remain; for example, we need to discover how to perform these tasks:

- Select a set of sliders from a large set of attributes.
- Specify greater than, less than, or greater than and less than.
- Deal with Boolean combinations of slider settings.
- Choose among highlighting by color, by points or light, by regions, by blinking, and so on.
- Cope with tens of thousands of points.
- Permit weighting of criteria.

The dynamic-query approach to the chemical table of elements was tested in an empirical comparison with a form-fillin query interface (Ahlberg et al., 1991). The counterbalanced-ordering within-subjects design with 18 chemistry students showed strong advantages for the dynamic queries, in terms of faster performance and lower error rates (Ahlberg et al., 1991).

Commercial information-retrieval systems, such as DIALOG or First-Search, permit complex Boolean expressions with parentheses, but their widespread adoption has been inhibited by their difficulty of use. Numerous proposals have been put forward to reduce the burden of specifying complex Boolean expressions (Reisner, 1988). Part of the confusion stems from informal English usage, in which a query such as "List all employees who live in New York and Boston" usually would result in an empty list because the "and" would be interpreted as an intersection; only employees who live in both cities would qualify! In English, "and" usually expands the options; in Boolean expressions, AND is used to narrow a set to the intersection of two others. Similarly, in the English "I'd like Russian or Italian salad dressing," the "or" is exclusive, indicating that you want one or the other but not both; in Boolean expressions, an OR is inclusive, and is used to expand a set.

The desire for *full Boolean expressions*, including nested parentheses and NOT operators, has led to novel metaphors for query specification. *Venn diagrams* (Michard, 1982) and *decision tables* (Greene et al., 1990) have been used, but these representations become clumsy as query complexity increases. To support arbitrarily complex Boolean expressions with a graphical specification, we applied the metaphor of water flowing from left to right through a series of filters, where each filter lets through only the appropriate documents, and the flow paths indicate AND or OR (Young and Shneiderman, 1993).

In this filter–flow model, ANDs are shown as a linear sequence of filters, suggesting the successive application of required criteria. As the flow passes through each filter, it is reduced, and the visual feedback shows a narrower stream of water. In Fig. 15.18(a) a journal database containing 6741 articles passes through the Date filter, about one-half of the articles satisfy the Date requirements of 94 to 97 (years 1994 to 1997). Only about one quarter of those articles pass through the Language filter, which selects English OR French. Users can also specify ORs across attributes, by putting filters in parallel flow paths (Fig. 15.18b). When the parallel flow paths converge, the width reflects the size of the union of the document sets.

Negation is handled by a NOT operator that, when selected, inverts all currently selected items in a filter (Fig. 15.18b). In the example, NOT 91 allows about 80 percent of the articles to pass the Date filter. Clusters of filters and flow paths (with one ingoing and one outgoing flow) can be made into a single labeled filter. Creation of clusters ensures that the full query can be shown on the display at once, and allows named clusters to be saved in a library for later reuse.

The filter-flow approach has been shown to help novices and intermittent users to specify complex Boolean expressions and to learn Boolean concepts. A usability study was conducted with 20 subjects who had had little experience using Boolean algebra. The prototype filter-flow interface was preferred over textual interface by all 20 subjects, and statistically significant advantages emerged on comprehension and composition tasks.

Another form of filtering is to apply a user-constructed set of keywords to dynamically generated information, such as incoming electronic-mail mes-

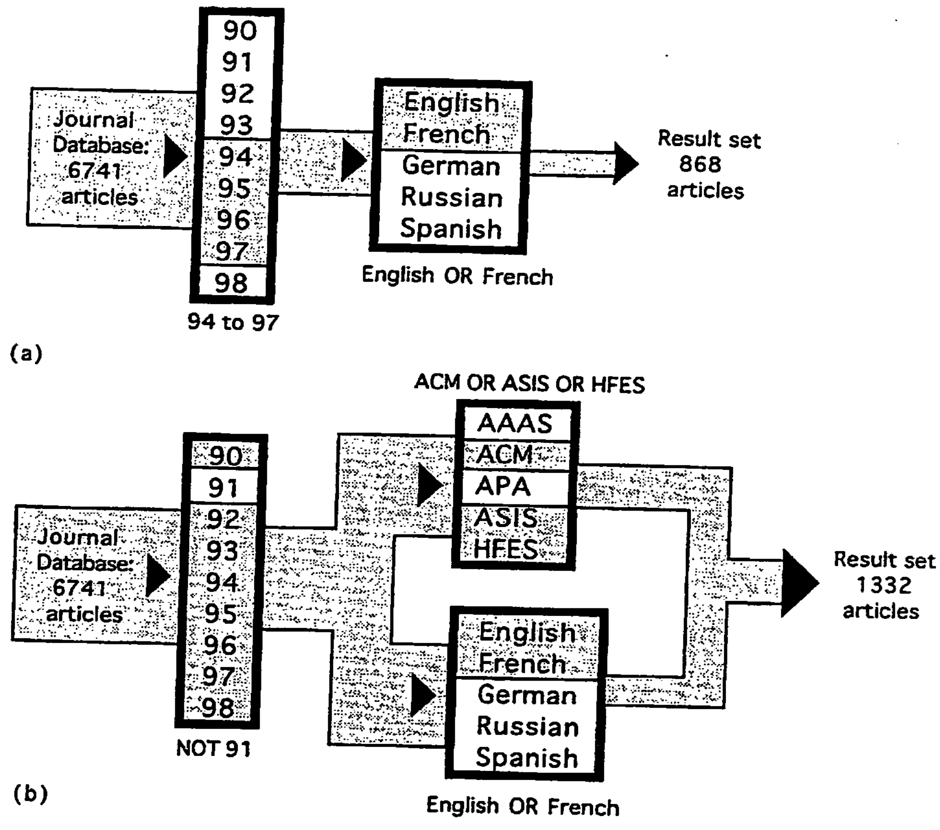


Figure 15.18

(a) Filter-flow model for the query (Date between 94 to 97) AND (Language is English OR French).
(b) Filter-flow model for query (Date NOT 91) AND (Publisher is ACM OR ASIS OR HFES) OR (Language is English OR French)).

sages, newspaper stories, or scientific journal articles (Belkin and Croft, 1992). The users create and store their profiles, which are evaluated each time that a new document appears. Users can be notified by electronic mail that a relevant document has appeared, or the results can be simply collected into a file until the users seek them out. These approaches are a modern version of traditional information-retrieval strategy called *selective dissemination of information (SDI)*, which was used in the earliest days of magnetic-tape distribution of document collections. Elaborate strategies for using the user-supplied set of keywords include latent semantic indexing, use of thesauri for find narrower or broader terms, and natural-language parsing techniques (Foltz and Dumais, 1992). Use of these strategies and term-frequency data can produce relevance rankings of retrieved documents that are appealing to many users and are successful in increasing the recall and precision of searches. A series of text-retrieval conferences (TREC) organized by Donna Harman at the National Institute for Standards and

Technology (http://potomac.ncsl.nist.gov/TREC) has allowed developers of research and commercial products to compare their strategies against a large test collection of textual documents.

A social form of filtering is *collaborative filtering,* in which groups of users combine their evaluations to help one another find interesting items in a large document collection (Resnick et al., 1994). Each user rates documents in terms of their interest. Then, the system can suggest unread articles that are close to the user's interests, as determined by matches with other people's interests. This method can also be applied to movies, music, restaurants, and so on. For example, if you rate six restaurants high, the algorithms will provide you with other restaurants that were rated high by people who liked your six restaurants. This strategy has an inherent appeal, and dozens of systems have been built for organizational databases, news files, music groups, and World Wide Web pages.

15.6 Practitioner's Summary

Improved user interfaces to traditional database-query and text- or multimedia-document search will spawn appealing new products. Flexible queries against complex text, sound, graphics, image, and video databases are emerging. Novel graphical and direct-manipulation approaches to query formulation and information visualization are now possible. Whereas research prototypes have typically dealt with only one data type (one-, two-, and three-dimensional data; temporal and multidimensional data; and tree and network data), successful commercial products will have to accommodate several. These products will need to provide smooth integration with existing software and to support the full task list: overview, zoom, filter, details-on-demand, relate, history, and extract. These methods are attractive because they present information rapidly and allow user-controlled exploration. If they are all to be fully effective, we will require advanced data structures, high-resolution color displays, fast data retrieval, and novel forms of user training. Many user interfaces for specifying advanced filtering are being built and are worthy of evaluation for commercial projects.

15.7 Researcher's Agenda

Although the computer contributes to the information explosion, it is potentially the magic lens for finding, sorting, filtering, and presenting the relevant items. Search in complex structured documents, graphics, images, sound, or video presents grand opportunities for the design of advanced user interfaces and powerful search engines to find the needles in the haystacks and the forests beyond the trees. The novel information-exploration tools—such as dynamic queries, treemaps, fisheye views, parallel coordinates, starfields, and perspective walls—are but a few of the inventions that will have to be tamed and validated by user-interface researchers. A better integration with perceptual psychology (understanding preattentive processes and the impact of varied coding or highlighting techniques) and with business decision making (identifying tasks and procedures that occur in realistic situations) is needed, as are theoretical foundations and practical benchmarks for choosing among the diverse emerging visualization techniques. Empirical studies would help to sort out the specific situations in which visualization was most helpful. Finally, software toolkits for building innovative visualizations would facilitate the exploration process.

World Wide Web Resources

WWW

The search services such as Alta Vista, Excite, Infoseek, and Lycos provide remarkable but flawed access to the World Wide Web. Other information retrieval topics such as collaborative filtering, document summarization, and indexing methods are covered. Information visualization tools are growing more effective for a wider range of tasks.

http://www.aw.com/DTUI

References

Ahlberg, Christopher and Shneiderman, Ben, Visual information seeking: Tight coupling of dynamic query filters with starfield displays, *Proc. CHI '94 Conference: Human Factors in Computing Systems*, ACM, New York (1994), 313–321 and color plates.

Ahlberg, Christopher and Shneiderman, Ben, AlphaSlider: A compact and rapid selector, *Proc. of ACM CHI '94 Conference Human Factors in Computing Systems*, ACM, New York (1994), 365–371.

Ahlberg, Christopher, Williamson, Christopher, and Shneiderman, Ben, Dynamic queries for information exploration: An implementation and evaluation, *Proc. ACM CHI '92: Human Factors in Computing Systems*, ACM, New York (1992), 619–626.

Ahlberg, Christopher and Wistrand, Erik, IVEE: An information visualization and exploration environment, *Proc. IEEE Information Visualization '95*, IEEE Computer Press, Los Alamitos, CA (1995), 66–73.

Andrews, Keith, Visualising cyberspace: Information visualisation in the Harmony internet browser, *Proc. IEEE Information Visualization '95*, IEEE Computer Press, Los Alamitos, CA (1995), 97–104.

Asahi, T., Turo, D., and Shneiderman, B., Using treemaps to visualize the analytic hierarchy process, *Information Systems Research*, 6, 4 (December 1995), 357–375.

Bartram, Lyn, Ho, Albert, Dill, John, and Henigman, Frank, The continuous zoom: A constrained fisheye technique for viewing and navigating large information spaces, *Proc. User Interface Software and Technology '95*, ACM, New York (1995), 207–215.

Becker, Richard A., Eick, Stephen G., and Wilks, Allan R. Visualizing network data, *IEEE Transactions on Visualization and Computer Graphics*, 1, 1 (March 1995), 16–28.

Bederson, Ben B. and Hollan, James D., PAD++: A zooming graphical user interface for exploring alternate interface physics, *Proc. User Interfaces Software and Technology '94* (1994), 17–27.

Belkin, Nick J. and Croft, Bruce W. Information filtering and information retrieval: Two sides of the same coin?, *Communications of the ACM*, 35, 12 (1992), 29–38.

Bertin, Jacques, *Semiology of Graphics*, University of Wisconsin Press, Madison, WI (1983).

Borgman, Christine, L., Why are online catalogs hard to use? Lessons learned from information-retrieval studies, *Journal of the American Society for Information Science*, 37, 6 (1986), 387–400.

Card, Stuart K., Robertson, George G., and York, William, The WebBook and the WebForager: An information workspace for the World-Wide Web, *Proc. CHI' 96 Conference: Human Factors in Computing Systems*, ACM, New York (1996), 111–117.

Carriere, Jeremy and Kazman, Rick, Interacting with huge hierarchies: Beyond cone trees, *Proc. IEEE Information Visualization '95*, IEEE Computer Press, Los Alamitos, CA (1995), 74–81.

Chimera, Richard, Value bars: An information visualization and navigation tool for multiattribute listings, *Proc. CHI '92 Conference: Human Factors in Computing Systems*, ACM, New York (1992), 293–294.

Chimera, Richard and Shneiderman, Ben, Evaluating three user interfaces for browsing tables of contents, *ACM Transactions on Information Systems*, 12, 4 (October 1994), 383–406.

Chuah, Mei C., Roth, Steven F., Mattis, Joe, and Kolojejchcik, John, SDM: Malleable Information Graphics, *Proc. IEEE Information Visualization '95*, IEEE Computer Press, Los Alamitos, CA (1995), 66–73.

Cleveland, William, *Visualizing Data*, Hobart Press, Summit, NJ (1993).

Egan, Dennis E., Remde, Joel R., Gomez, Louis M., Landauer, Thomas K., Eberhardt, Jennifer, and Lochbum, Carol C., Formative design-evaluation of SuperBook, *ACM Transactions on Information Systems*, 7, 1 (January 1989), 30–57.

Egenhofer, Max and Richards, J., Exploratory access to geographic data based on the map-overlay metaphor, *Journal of Visual Languages and Computing*, 4, 2 (1993), 105–125.

Eick, Stephen G., Steffen, Joseph L., and Sumner, Jr., Eric E., SeeSoft: A tool for visualizing line-oriented software statistics, *IEEE Transactions on Software Engineering*, 18, 11 (1992) 957–968.

Eick, Stephen G. and Wills, Graham J., Navigating large networks with hierarchies, *Proc. IEEE Visualization '93 Conference* (1993), 204–210.

Fairchild, Kim M., Poltrock, Steven E., and Furnas, George W., SemNet: Three-dimensional representations of large knowledge bases. In Guindon, Raymonde (Editor), *Cognitive Science and its Applications for Human–Computer Interaction*, Lawrence Erlbaum, Hillsdale, NJ (1988), 201–233.

Fishkin, Ken and Stone, Maureen C., Enhanced dynamic queries via movable filters, *Proc. CHI' 95 Conference: Human Factors in Computing Systems*, ACM, New York (1995), 415–420.

Foltz, Peter W. and Dumais, Susan T., Personalized information delivery: An analysis of information filtering methods. *Communications of the ACM*, 35, 12 (1992), 51–60.

Furnas, George W., Generalized fisheye views, *Proc. CHI' 86 Conference: Human Factors in Computing Systems*, ACM, New York (1986), 16–23.

Goldstein, Jade and Roth, Steven F., Using aggregation and dynamic queries for exploring large data sets, *Proc. CHI' 95 Conference: Human Factors in Computing Systems*, ACM, New York (1995), 23–29.

Greene, S. L., Devlin, S. J., Cannata, P. E., and Gomez, L. M., No IFs, ANDs, or ORs: A study of database querying, *International Journal of Man–Machine Studies*, 32, (March 1990), 303–326.

Hendley, R. J., Drew, N. S., Wood, A. S., Narcissus: Visualizing information, *Proc. IEEE Information Visualization '95*, IEEE Computer Press, Los Alamitos, CA (1995), 90–96.

Humphrey, Susanne M. and Melloni, Biagio John, *Databases: A Primer for Retrieving Information by Computer*, Prentice-Hall, Englewood Cliffs, NJ (1986).

Inselberg, Alfred, The plane with parallel coordinates, *The Visual Computer*, 1, (1985), 69–91.

Jarke, M., and Vassiliou, Y., A framework for choosing a database query language, *ACM Computing Surveys*, 11, 3 (1986), 313–340.

Jerding, Dean F. and Stasko, John T., The information mural: A technique for displaying and navigating large information spaces, *Proc. IEEE Information Visualization '95*, IEEE Computer Press, Los Alamitos, CA (1995), 43–50.

Jog, Ninad and Shneiderman, Ben, Information visualization with smooth zooming on an starfield display *Proc. Visual Databases 3* (March 1995), 1–10.

Johnson, Brian, and Shneiderman, Ben, Tree-maps: A space-filling approach to the visualization of hierarchical information structures, *Proc. IEEE Visualization '91*, IEEE, Piscataway, NJ (1991), 284–291.

Keim, D. A. and Kriegal, H., VisDB: Database exploration using multidimensional visualization, *IEEE Computer Graphics and Applications* (September 1994), 40–49.

Kim, H. J., Korth, H. F., and Silberschatz, A., PICASSO: A graphical query language, *Software: Practice and Experience*, 18, 3 (1988), 169–203.

Koenemann, Juergen and Belkin, Nicholas, A case for interaction: A study of interactive information retrieval behavior and effectiveness, *Proc. CHI '96 Human Factors in Computing Systems*, ACM Press, New York (1996), 205–212.

Korfhage, Robert, To see or not to see: Is that the query?, *Communications of the ACM*, 34 (1991), 134–141.

Kumar, Harsha, Plaisant, Catherine, and Shneiderman, Ben, Browsing hierarchical data with multi-level dynamic queries and pruning, *International Journal of Human–Computer Studies*, 46, 1 (January 1997), 103–124.

Lamping, John, Rao, Ramana, and Pirolli, Peter, A focus + context technique based on hyperbolic geometry for visualizing large hierarchies, *Proc. of CHI '95*

Conference: Human Factors in Computing Systems, ACM, New York (1995), 401–408.

Laurini, R. and Thompson, D., *Fundamentals of Spatial Information Systems*, Academic Press, New York (1992).

Marchionini, Gary, *Information Seeking in Electronic Environments*, Cambridge University Press, UK (1995).

Marchionini, Gary and Shneiderman, Ben, Finding facts and browsing knowledge in hypertext systems, *IEEE Computer*, 21, 1 (January 1988), 70–80.

Mark, Leo, A graphical query language for the binary relationship model, *Information Systems*, 14, 3 (1989), 231–246.

McCormick, B., DeFanti, T, and Brown, R. (Editors), Visualization in scientific computing and computer graphics, *ACM SIGGRAPH*, 21, 6 (November 1987).

Michard, A., A new database query language for non-professional users: Design principles and ergonomic evaluation, *Behavioral and Information Technology*, 1, 3 (July–September 1982), 279–288.

Mitchell, Richard, Day, David, and Hirschman, Lynette, Fishing for information on the internet, *Proc. IEEE Information Visualization '95*, IEEE Computer Press, Los Alamitos, CA (1995), 105–111.

Mukherjea, Sougata, Foley, James D., and Hudson, Scott, Visualizing complex hypermedia networks through multiple hierarchical views, *Proc. of ACM CHI '95 Conference: Human Factors in Computing Systems*, ACM, New York (1995), 331–337 plus color plate.

North, Chris, Shneiderman, Ben, and Plaisant, Catherine, User controlled overviews of an image library: A case study of the Visible Human, *Proc. 1st ACM International Conference on Digital Libraries* (1996), 74–82.

Pirolli, Peter, Schank, Patricia, Hearst, Marti, and Diehl, Christine, Scatter/gather browsing communicates the topic structure of a very large text collection, *Proc. of ACM CHI' 96 Conference*, ACM, New York (1996), 213–220.

Plaisant, Catherine, Rose, Anne, Milash, Brett, Widoff, Seth, and Shneiderman, Ben, LifeLines: Visualizing personal histories, *Proc. of CHI' 96 Conference: Human Factors in Computing Systems*, ACM, New York (1996), 221–227, 518.

Reisner, Phyllis, Query languages. In Helander, Martin (Editor), *Handbook of Human–Computer Interaction*, North-Holland, Amsterdam, The Netherlands (1988), 257–280.

Rao, Ramana and Card, Stuart K., The Table Lens; Merging graphical and symbolic representations in an interactive focus + context visualization for tabular information, *Proc. CHI '94 Conference: Human Factors in Computing Systems*, ACM, New York (1994), 318–322.

Resnick, Paul, Iacovou, Neophytos, Suchak, Mitesh, Bergstrom, Peter and Riedl, John, GroupLens: An open architecture for collaborative filtering of netnews, *Proc. Conference on Computer Supported Cooperative Work '94*, ACM, New York (1994), 175–186.

Robertson, George G., Card, Stuart K., and Mackinlay, Jock D., Information visualization using 3-D interactive animation, *Communications of the ACM*, 36, 4 (April 1993), 56–71.

Robertson George G. and Mackinlay, Jock D., The document lens, *Proc. 1993 ACM User Interface Software and Technology*, ACM New York (1993), 101–108.

Roth, Steven F., Lucas, Peter, Senn, Jeffrey A., Gomberg, Cristina C., Burks, Michael B., Stroffolino, Philip J., Kolojejchick, John A. and Dunmire, Carolyn, Visage: A user interface environment for exploring information, *Proc. IEEE Information Visualization '96*, IEEE Computer Press, Los Alamitos, CA (1996), 3–12.

Salton, G., Automatic Text Processing: *The Transformation, Analysis, and Retrieval of Information by Computer*, Addison-Wesley, Reading, MA (1989).

Sarkar, Manojit and Brown, Marc H., Graphical fisheye views, *Communications of the ACM*, 37, 12 (July 1994), 73–84.

Schaffer, Doug, Zuo, Zhengping, Greenberg, Saul, Bartram, Lyn, Dill, John, Dubs, Shelli and Roseman, Mark, Navigating hierarchically clustered networks through fisheye and full-zoom methods, *ACM Transactions on Computer–Human Interaction*, 3, 2 (June 1996), 162–188.

Shneiderman, Ben, Tree visualization with tree-maps: A 2-D space-filling approach, *ACM Transactions on Graphics*, 11, 1 (January 1992), 92–99.

Shneiderman, Ben, Dynamic queries for visual information seeking, *IEEE Software*, 11, 6 (1994), 70–77.

Shneiderman, Ben, Brethauer, Dorothy, Plaisant, Catherine and Potter, Richard, Three evaluations of museum installations of a hypertext system, *Journal of the American Society for Information Science*, 40, 3 (May 1989), 172–182.

Shneiderman, Ben, Byrd, Donald, and Croft, Bruce, Clarifying search: A user-interface framework for text searches, *D-LIB Magazine of Digital Library Research* (January 1997), http://www.dlib.org/.

Spence, Robert and Apperley, Mark, Data base navigation: An office environment for the professional, *Behaviour & Information Technology*, 1, 1 (1982), 43–54.

Spoerri, Anslem, InfoCrystal: A visual tool for information retrieval and management, *Proc. ACM Conf. on Information and Knowledge Management* (1993), 150–157.

Tufte, Edward, *The Visual Display of Quantitative Information*, Graphics Press, Cheshire, CT (1983).

Tufte, Edward, *Envisioning Information*, Graphics Press, Cheshire, CT (1990).

Tweedie, Lisa, Spence, Robert, Dawkes, Huw, and Su, Hua, Externalising abstract mathematical models, *Proc. of CHI' 96 Conference: Human Factors in Computing Systems*, ACM, New York (1996), 406–412.

Weiland, William J. and Shneiderman, Ben, A graphical query interface based on aggregation/generalization hierarchies,, *Information Systems*, 18, 4 (1993), 215–232.

Welty, C., Correcting user errors in SQL, *International Journal of Man–Machine Studies*, 22 (1985), 463–477.

Williamson, Christopher, and Shneiderman, Ben, The Dynamic HomeFinder: Evaluating dynamic queries in a real-estate information exploration system, *Proc. ACM SIGIR '92 Conference*, ACM, New York (1992), 338–346. Reprinted in Shneiderman, B. (Editor), *Sparks of Innovation in Human–Computer Interaction*, Ablex Publishers, Norwood, NJ (1993), 295–307.

Wise, James A., Thomas, James, J., Pennock, Kelly, Lantrip, David, Pottier, Marc, Schur, Anne, and Crow, Vern, Visualizing the non-visual: Spatial analysis and interaction with information from text documents, *Proc. IEEE Information Visualization '95*, IEEE Computer Press, Los Alamitos, CA (1995), 51–58.

Wurman, Richard Saul, *Information Anxiety*, Doubleday, New York (1989).

Young, Degi and Shneiderman, Ben, A graphical filter/flow model for Boolean queries: An implementation and experiment, *Journal of the American Society for Information Science*, 44, 6 (July 1993), 327–339.